USING THE DEEP GRAPH REPRESENTATION LEARNING METHOD TO STUDY THE INDIAN ELECTION SOCIAL NETWORK

By

Yimo Liu

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Business Administration - Business Information Systems - Doctor of Philosophy
Computational Mathematics, Science and Engineering - Dual Major

2022

ABSTRACT

USING THE DEEP GRAPH REPRESENTATION LEARNING METHOD TO STUDY THE INDIAN ELECTION SOCIAL NETWORK

By

Yimo Liu

The aim of this dissertation is using the Deep Graph Learning to study on two problems in social network data. This dissertation includes two essays. The first essay is using Deep Graph Representation Learning to solve the counterfactual inference problem on social network. The second essay is using Deep Graph Representation Learning to solve the fairness recommendation results on social network.

Graph Representation Learning is very useful method to extract graph features from the large-scale network. There are many advantages when we use this method to map the data from the graph space into the embedding space. First, there is no need to use the predefined measurement on graph structure such as bipartite graph, tripartite graph, etc, but directly keep the information of the network topology as good as possible. Second, when we study on social network platform such as Facebook, Twitter, etc, we need to consider more attributes with nodes and edges. Therefore, the information in the social network becomes more complicated. Considering these two advantages, we choose the graph representation learning to embed these features into the representation space.

The first essay is combining this method with the transfer learning in order to fit the Rubin's counterfactual framework. It can help to give a robust and lowest biased estimation results compared with the propensity score matching methods.

The second essay is borrowing the advantage of the graph representation learning method to learn the representation space of different types of the users and connections in twitter. We

combine this method with the attention mechanism to construct the fairness based loss function. This can help to increase the fairness of the recommendation and maintain the predicted accuracy of the recommendation in social network.

ACKNOWLEDGEMENTS

First, I want to thank the advisor of the dissertation committee, Professor Anjana Susarla. Under her guidance, I choose this graph neural network related methods as the dissertation topics. Both causal inference problem and recommendation fairness problem are very popular in academic research. These two questions have broad audience in many disciplines. I also sincerely thank other committee members, Professor Quan Zhang, Professor Yuying Xie, and Professor Matthew Hirn. Their suggestions includes both the model analysis and the empirical analysis. It helps me to figure out many tricky and difficult parts of this research projects. Considering that my dissertation is for satisfying on both Information Systems and Computational Mathematics disciplines, their suggestions are always helpful and considerable. I also want to regard the Professor Yingda Lu and Professor Nishtha Langer to support this rich dataset and give the nicely suggestions.

Second, it was lucky and honor to have chance to join both the business school and the engineering school for the last six years at Michigan State University. I learned a lot of advanced knowledge and technique skills in the courses, lectures, and workshops. I saw how world class scholars discussed, critiqued, and created the academic researches. It is really helpful for me to continue my research job in the future career.

Third, thank my family, my friends in Lansing, and my girlfriend, who always stand behind me and support me with their love and time. We played basketball together, had meal together, and shared happiness and sadness together. I cannot get any achievement without your help.

In the end, thank God to let me still alive.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

ESSAY 1. A GRAPH REPRESENTATION LEARNING METHOD FOR MATCHING THE ONLINE POLITICAL SUPPORTERS ON THE SOCIAL NETWORK

0. ABSTRACT OF ESSAY 1

Twitter is one of the most important platforms for politicians to communicate with their supporters. However, the causal mechanism behind how politicians use the twitter as an electronic campaign tool to impact their supporters is few researched. This paper studies how politicians post tweets and impact their supporters' sentiments on the twitter. In particular, we focus on study the political hashtags causally affect on people's reactions on twitter. We use the 2014 Indian general election data to construct the retweets network and detect the causal inference based on this retweets network. The most difficult part of detecting the causal inference problem is avoiding confoundedness factors. However, the social network data directly break the assumption of constructing the propensity scores based method, which leads to the bias estimator. In this paper, we use the deep graph representation learning to construct a better matching method under the social network circumstance. We find that, compared with propensity score based method, our method can always give the lowest error result under balanced covariates and small-scale, imbalanced covariates and small-scale, balanced covariates and large-scale, imbalanced covariates and large-scale social network circumstances.

1. INTRODUCTION

Social network plays an important role in our lives. There are a lot of people communicate and discuss the political events on the social network platforms such as Facebook, Reddit, and Twitter, etc.[1] This in turn attracts many politicians and parties use these social network platforms as tool to present themselves and campaign. For example, President Obama used twitter to interact with his voters. Even his budget is lower than Hillary's budget, President Obama still won the election in 2008. Some researchers account his winning to the success of using twitter. He is very good at using twitter to interact with his voters during the presidential campaign. Except posting more interesting and attractive contents in order to encourage their followers to engage the politics, politicians also analysis the sentiments of the followers and modify their policies based on their reaction. They need to analyze the reason behind the successful tweets and modify their strategies on target voters. (Ahmed et al 2016, Khan et al 2020, Panda et al 2020)

Exploring the reason behind the successful tweets, we want to identify the causal effects from the opinion leader's tweet. For example, if the president Trump posts a tweet as "With what I am doing in the fight with the Drug Companies, drug prices will be coming down 50, 60, and even 70 percent.". This post may influence the sentiments from the followers for the following reasons:

1) The follower gives a positive or negative comments because he or she likes or dislikes the drug prices got decreased.

2) The follower gives a positive or negative comment because he or she supports or not supports Trump no matter what he tweets.

[1] https://www.pewresearch.org/fact-tank/2019/08/02/10-facts-about-americans-and-twitter/

3) The follower retweets because he or she gets affected from the neighbors' sentiments on twitter.

The last two reasons are most common confounding factors when we detect the causal effect under social network circumstance (Fang et al 2013, Ma et al 2015, Zhang et al 2018, Rishika and Ramaprasad 2019, Gelper et al 2020). The second reason is called homophily, which means users tend to give positive sentiments to the opinion leader because they have the similar demographic information such as age, gender, etc (McPherson et al 2001, Krivitsky et al 2009). The third reason is called peer influence. It means that, users are easy to get influenced from other connected users on the social network (Lewis 2012, Christakis and Fowler 2013). If most of other users give positive or negative sentiments on the Trump's tweet, one tends to give sentiments similar to his or her neighbors. If the opinion leaders plan to please their base voters, attract swing voters and new voters based on their reactions on the tweets, they need to distinguish the first reason from the second reason and the third reason. To detect the homophily and peer influence are very important to estimate the causal effect behind the successful tweets. (Shalizi and Thomas 2011, Ma et al 2015 , Zhang et al 2018, Gelper et al 2020)

Many studies have focused on separating homophily and peer influence from the causal inference under non-social-network circumstance based on the Rubin's (2004) framework, the propensity score matching method. (Hirano and Imbens 2004, Aral et al 2009, Imai and Ratkovic 2014, Forastiere et al 2020, Sant'Anna et al 2020) This method requires the treated group and the control group to follow the independent identical distribution condition (also called SUVTA assumption) and both the treated group and the control group outcomes should be independent with the treatment (also called unconfounded assumption). If we can construct the balanced propensity score based on the covariate samples, which means there is no significant difference

between treated group and the control group, we can match the similar units on both treated group and control group based on their propensity scores. (Rubin 2004, Hirano and Imbens 2004) The difference between their outcomes is the treatment effect. This method is very success under the circumstance that the covariate sample sets is small and balanced and the relationship between covariates and outcomes are linear (Johansson et al 2016, Veitch 2019). However, the propensity score matching method is not very efficient under the social network circumstance (Tchetgen and VanderWeele 2012, Huffman and Gameren 2018). The network data directly break these two assumptions because social tie and homophily exist. For example, some users have a political preference on republican and some others have a political preference on democrats. If we choose the users have a political preference on republican as the treated group and the users have a political preference on democrats as the control group and try to test how one tweet affects their opinions. Both treated group users and control group users can receive the same message from the opinion leader, we cannot separate them very clearly. This situation breaks the SUVTA assumption and leads to overestimates the causal effect on the network (Forastiere et al 2020, Jackson et al 2020). Statisticians solve these problems by modifying the propensity score with information from the confounders, or the network structure. Hirano and Imbens (2004) build a propensity score by generalized the un-confoundedness to solve the homopily problem. Jackson et al (2020) solve the peer influence problem based on generalized propensity score including the local graph structure. These methods can somewhat help to avoid overestimate the causal effect on the social network platform. However, it is still difficult to construct the balanced propensity scores on the social network (King and Nielsen 2019).

We address the casual inference problem under social network circumstance by using the graph-based representation learning. It can transform the covariate sample sets into a latent space, also called representation space, and match the treatment group and control group into a balanced representation space by a kernel statistical test of independence. In the meantime, this method also calculate the relationship between covariates and outcome so that it can handle more nonlinear relationship between covariates and outcome. The generalized error of the causal effect based on the representation learning method are bonded by the math proofs. (Johansson et al 2016, Li and Fu 2017, Kallus 2017, Kallus 2020)

The advantage of our model is capturing more unobserved covariates, which implicit includes the information about network structure, latent homophily, etc. We use the node embedding method to represent the node, edge and attributes information (Grover and Leskovec 2016). These information can represent the homophily and peer influence. We learn the representation space based on these network related information, it helps to calculate the balanced covariate metric and capture the nonlinear relationship between covariate variable and the outcome. The deep neural network architecture can help to calculate the large and nonlinear representation space easily.

We use the data with Indian election 2014 and compare our model with the propensity score matching method under four types of dataset. The small and covariate balanced case, the large and covariate balanced case, the small and covariate imbalanced case, the large and covariate imbalanced case. The results show that our graph representation learning method can always give lowest bias estimation on each case. If we want to test the casual inference under the social network data, the graph representation learning method is more robust and flexible than the propensity score based method.

The organization of our papers is: First, a literature review on causal inference under social network circumstance based on the propensity score matching method and based on the representation learning method. Second, a describing and building of graph neural network model. Third, how this graph representation based method causally explain the social influence problem. Fourth, the data result, and Fifth the discussion.

2. LITERATURE REVIEW

2.1. Propensity Score Matching method under network data

The most general way to test the causal inference problem is using the Propensity Score Matching Method. In order to construct the propensity scores well, Rubin (1987) requires the covariate variables between the treated group and the control group should be balanced. The Propensity Score Matching Method can solve the selection bias problem under Rubin's two assumptions. However, the network data break the SUTVA assumption and no hidden confounder assumption. It leads to imbalanced covariates between the treated group and the control group. The treatment effect is overestimated under this situation. Many statistical researchers tried to solve these two problems by modifying the construction ways.

2.1.1. Solve hidden confounder problem

Hirano and Imbens (2004) are the pioneers to solve the hidden confounder problem. They build a propensity score by generalized the un-confoundedness: each treatment is independent with each outcome y_i conditional by different covariates but not for all outcome Y. This method can reduce the bias during continuous treatment situations. Imai and Ratkovic (2014) proposed a method that the treatment is independent with treatment conditional on inverse propensity score weighting moment. This method can help to justify the over-identify problem of the propensity score method. Huffman and Gameren (2018) applied this method to the continuous interventions case. They replace the weight by generalized method of moment. Sant'Anna et al (2020) follow the covariate balanced propensity score idea but just flexible the inverse weighting moment to integrated conditional moment. This method is quite close to our integral probability metric approach. However, this method cannot compute efficiently during high dimension covariates situation. Therefore, they did not apply this method on the network data but on employment data.

2.1.2. Solve Interference Problem

SUTVA assumption means the outcome of user A has no influence on the outcome of user B. It is not held under network data. For example, both user B and user C see a tweet given by user A on twitter. If user C investigates that the user B gives a negative comment on this tweet, user C may also give a negative comment because he or she gets affected by user B's comment. This situation always happened on social network and directly break the SUTVA assumption.

Some prior studies solve this problem by assuming these nodes are interference on the unit level but independent on the group level. They made some clusters on nodes at first step, and constructed the propensity score based on group level node. (Hudgens and Halloran 2008, Tchetgen and VanderWeele 2012, Liu L and Hudgens 2014, Arpino et al 2017) Some part of the total treatment effect is coming from the peer effect but not the treatment. Therefore, they extract the peer effect or named indirect causal effect from the total treatment effect to solve this interference problem.

Some recent studies are based on the general propensity score method to solve the node level interference problem. (Aronow and Samii 2017, Ogburn and VanderWeele 2017, Forastiere et al 2020, Jackson et al 2020). Aronow and Samii (2017), Ogburn and VanderWeele (2017) discussed the theory of the estimator consider whether a node gets exposed to neighbor nodes or not. After we use the total treatment effect minus the treatment effect, which including the neighbor nodes exposure, we can get the pure causal effect. Forastiere et al (2020) gave both theory and empirical analysis of this approach, and prove that we can get a better result.

Jackson et al (2020) solve this problem not based on generalized propensity score but based on constructing the propensity score based on the peer influence. They assume the graph is the fully

connected directed graph. Their propensity score is still biased because their graph topology is not very flexible.

Table 1. Literature review of Propensity Score Matching method under network data

Author	Model assumption	Confounder type	Propensity score or Estimator Construction
Hirano and Imbenes (2004)	$\mathrm{y_i}(0), \mathrm{y_i}(1) \perp \mathrm{t_i} \mid e(x_i, t_i)$	Hidden Confounder	Construct a propensity score by generalized the un-confoundness
Imai and Ratkovic (2014)	$\mathrm{y_i}(0), \mathrm{y_i}(1) \perp \mathrm{t_i} \mid w(x_i)$ where w(.) is the inverse weighting moment	Hidden Confounder	Construct a propensity score by inverse weighting moment
Huffman and Gameren (2018).	Same with above	Hidden Confounder	Construct a propensity score by inverse weighting moment
Sant'Anna et al (2020)	$\mathrm{y_i}(0), \mathrm{y_i}(1) \perp \mathrm{t_i} \mid I(x_i)$ where w(.) is the integrated conditional moment	Hidden Confounder	Construct a propensity score by integrated conditional moment
Hudgens. and Halloran (2008)	$\mathrm{y_i}(x,t) = \frac{\sum_{j=1}^{n} Y \cdot I_j[x_i, t]}{\sum_{j=1}^{n} I_j[x_i, t]}$ where $I[x,t]$ is the group received treatment	Group-level interference	Extract peer effect from total causal effect by group
Tchetgen and VanderWeele (2012)	Same with above	Group-level interference	Extract peer effect from total causal effect by group
Liu and Hudgens (2014)	Same with above	Group-level interference	Extract peer effect from total causal effect by group

Table 1. (cont'd)

Aronow and Samii (2017)	$y_i(x,t) = \frac{\sum_{j=1}^{n} Y \cdot G_j[x_i,t]}{\sum_{j=1}^{n} G_j[x_i,t]}$ where $G_j[x_i,t]$ is the neighbor graph received treatment	Node-level interference	Extract peer effect from total causal effect by neighbor graph
Ogburn and VanderWeele (2017).	Same with above	Node -level interference	Extract peer effect from total causal effect by neighbor graph
Forastiere et al (2020)	Same with above	Node -level interference	Extract peer effect from total causal effect by neighbor graph
Jackson et al (2020)	$y_i(0), y_i(1) \perp t_i \mid e_i(x,g)$ Where g is directed graph	Node-level interference	Construct a propensity scores by including graph

2.2. Representation Learning for Counterfactual Inference

Even there are many statisticians tried to construct a better propensity score by including the graph information, it is still not easy to represent the information of the user on the social network very well. In other way, we can use the representation learning method to represent the information of the user. It is an efficient way to represent the complex relationship between different users on the social network. Then, we put the representation learning method into the counterfactual inference framework. This can help to solve the confounding problem.

2.2.1. Kernel Space Matching Method for Counterfactual Inference

The propensity score matching method is for matching the treated group and the control group on the one dimension subspace. It uses the logistic model to transform the covariate variables into the propensity scores. If we can find the balanced covariate variables, we can construct the propensity score very well. However, in order to capture the counterfactual inference on the

social network, we need to construct a large, nonlinear subspace to balance the treated group and control group. In order to transform the nonlinear, imbalanced covariate variables into the balanced representation space, we need to use the kernel space matching method. The advantage for using kernel space matching method is we can use the kernel trick to represent the distance between any two different units from the sample space. (Li and Fu 2017, Zhang et al 2019, Chu et al 2020)

The basic logic is using the outcome Y and input X to learn the encoding transformation ϕ. This can help to transform the covariate X into the representation $\phi(X)$. Then, use the decoding transformation ψ to transform the representation $\phi(X)$ into the output Y. We can observe the treated group $\mathrm{X}_{\mathrm{treat}}$ and the factual outcome Y_1, the treated group with treatment. If the transformation ϕ and ψ can successfully map the $\mathrm{X}_{\mathrm{treat}}$ into the outcome Y_1, the similar units in the control group $\mathrm{X}_{\mathrm{control}}$ should also be mapped into the counterfactual outcome $\widehat{\mathrm{Y}_1}$. Their difference is the treatment effect from the treated group. We can use the same idea to construct the treatment effect from the control group.(Caliendo and Kopeinig 2008, Li and Fu 2017, Kallus 2017)

In order to find the similar units from the treated group and the control group, we require the sample of treated group and the control group are identical under the kernel space. We use the distance functions called mean maximum discrepancy and the Wasserstein 1 distance to measure the distance between the distribution of the treated group and the distribution of the control group in the kernel space. We expect their distances are as small as possible.

2.2.2. Deep Representation Learning for Counterfactual Inference

If we replace the kernel transformation with the neural network, it becomes easy to compute the appropriate encoding function. The counterfactual Inference problem can be exchange to a

domain adaptation problem. If the treated group and the control group are assigned treatment randomly, their distributions in the representation space should be identical. We can use the discrepancy distance to measure it.

Therefore, there are two main streams on this study: Some of them focus on constructing a good discrepancy distance estimation (Johansson et al 2018, Yao et al 2018, Yao et al 2019, Sharma 2020). A common way to construct the discrepancy distance are maximum mean discrepancy and Wasserstein 1 distance (Johansson et al 2018, Kallus 2020, Flato et al 2019). Both of them can measure the distance between two distributions in a Non-Euclidean distance topology. Our model follows this method to calculate the distance between the factual domain and the counterfactual domain. Yao et al (2018) and Yao et al (2019) constructed the distance between two sample domains by finding specific connections on different nodes, which gives us higher accuracy but loses some generality. Their studies is based on the network data. Sharma (2020) used the general propensity score to calculate the distance, which can give a similar performance with maximum mean discrepancy. This method is not only for network data but also for other data structures.

Some others focus on constructing a good encoder function (Kallus 2020, Du et al 2019, Flato et al 2019, Alaa 2017, Schwab et al 2019). It includes basic graph neural network, convolution network, attention network, and generative network. The last three methods are just advanced methods to compute encoder function quickly. Our model follows the basic graph neural network and compares its performance with convolution network and attention network method.

Alaa (2017) and Schwab et al (2019) used a different framework because they directly used propensity score as the dropout network and combine it together with the encoder function.

After we build the good encoder function and discrepancy distance function, we can focus on social network data. As we discussed before, the social network has two types of confound problem lead to bias estimation: homophily and peer influence. Guo et al [a] (2020) and Guo et al [b] (2020) studied the homophily problem on social network data. They embed the latent homophily variable into the representation function and get a lower bias result. Ma et al (2020) studied the peer influence problem. They used the equation of the whole graph treatment effect minus the equation of peer influence effect to avoid the peer influence bias.

Table 2. Literature review of Propensity Score Matching method under network data

Author	Loss Function Components	Encoder Function	Discrepancy Distance Measurement
Johansson et al (2018)	Treatment Prediction Loss + Discrepancy Loss	Deep Neural Network	Kernel Maximum Mean Discrepancy and Wasserstein 1 distance
Yao et al (2018)		Deep Neural Network	Position Dependent Deep Metric + Middle Point Distance Minimization
Yao et al (2019)		Deep Neural Network	Similarity Preserve
Sharma (2020)		Graph neural network	General propensity score
Kallus (2020)		Generative Network	Kernel Maximum Mean Discrepancy and Wasserstein 1 distance

Table 2. (cont'd)

Du et al (2019)	Treatment Prediction Loss +Discrepancy Loss+ Mutual Information Loss	Generative Network	Mutual Information Estimator
Flato et al (2019)		Generative Network	Kernel Maximum Mean Discrepancy
Alaa (2017)		Deep Neural Network	Propensity Dropout Network
Schwab et al (2019)		Deep Neural Network	Propensity Dropout Network
Guo et al [a] (2020)		Graph Convolution Network	Wasserstein 1 Distance
Guo et al [b] (2020)		Graph Attention Mechanism	Inverse Propensity Scoring Weights
Ma et al (2020)		Graph Convolution Network	Kernel Maximum Mean Discrepancy and Wasserstein 1 Distance

2.2.3. Compare propensity score method with representation learning method

Comparing the propensity score method and representation learning method, we can find that if we don't have rich covariate data, we need to choose the representation learning method. If we have rich covariate data, we should choose the propensity score-based method. The propensity score-based method requires constructing the propensity score to calculates the difference between the treatment group and the control group at first and calculate the treatment effect from

the first step. The representation learning method calculate the treatment effect and the difference between treatment group and control group simultaneously.

Because there are two types of bias in the network data: hidden confounder and peer influence effect, the propensity score method needs to observe covariate variables as many as we can. We can construct the propensity score from the traditional network theory such as tie level, node level, and graph structure level. It give us more explanation on causal mechanism. But we cannot always expect there is no big difference between the treatment group and the control group. They may have significantly different kinds of Centrality, Density, and Closeness or have significantly different kinds of tie strength or tie directions between nodes. There are a lot of network features we cannot control, which leads to a bias propensity score or a propensity score without enough network information.

Therefore, using the representation learning method has a big advantage on solving network data because graph topology includes the information of hidden confounders and peer influence. It estimates the treatment effect simultaneously. As long as we can find an appropriate representation function, we can always calculate the treatment effect with the smallest bias. How to make sure we can find an appropriate representation function? An encoder-decoder framework can help to find the right function. This encoder-decoder framework is a general way to classify the data in machine learning research field. It can perfectly extend to the neural network.

Table 3. Compare Propensity Score Method and Representation Learning Method

	Pros	Cons
Propensity score method	Propensity Score Method has the advantage on learning linear, rich observed covariates data	X should be balanced (no significant differences) between the treatment group and the control group.
	Propensity Score can give global explanation above covariates data	Propensity Score is difficult to compute if covariate sample space is very large
Representation learning method	X doesn't need balanced on treatment group and control group	Representation Learning method cannot explain covariates but on node-level.
	Representation Learning is easy to compute if covariate sample space is very large	

Our model is combing the propensity score method and graph representation method to get both of their benefits. We construct the treatment group and control group from twitter and compare these two methods. Because the Propensity score method requires the covariate variables as rich as possible, we construct the covariate variables includes user-level features, node-level features, and network structure features. We build the propensity dropout network to replace the discrepancy distance function and still maintain the graph neural network to embed the whole graph structure.

Table 4. Compare our method with other methods

	Linear relation between X and Y	Nonlinear relation between X and Y	Graphical Nonlinear relation between X and Y
Rich Covariate balance	Propensity Score based method	Covariate Balanced Propensity Score	Our model Graph Representation Function+Discrepancy Distance Measurement+Distance between text information
Poor Covariate balance	Representation Function + Discrepancy Distance Measurement	NN+ Discrepancy Distance Measurement	GNN+Discrepancy Distance Measurement+Distance between text information

3. MODEL BUILDING

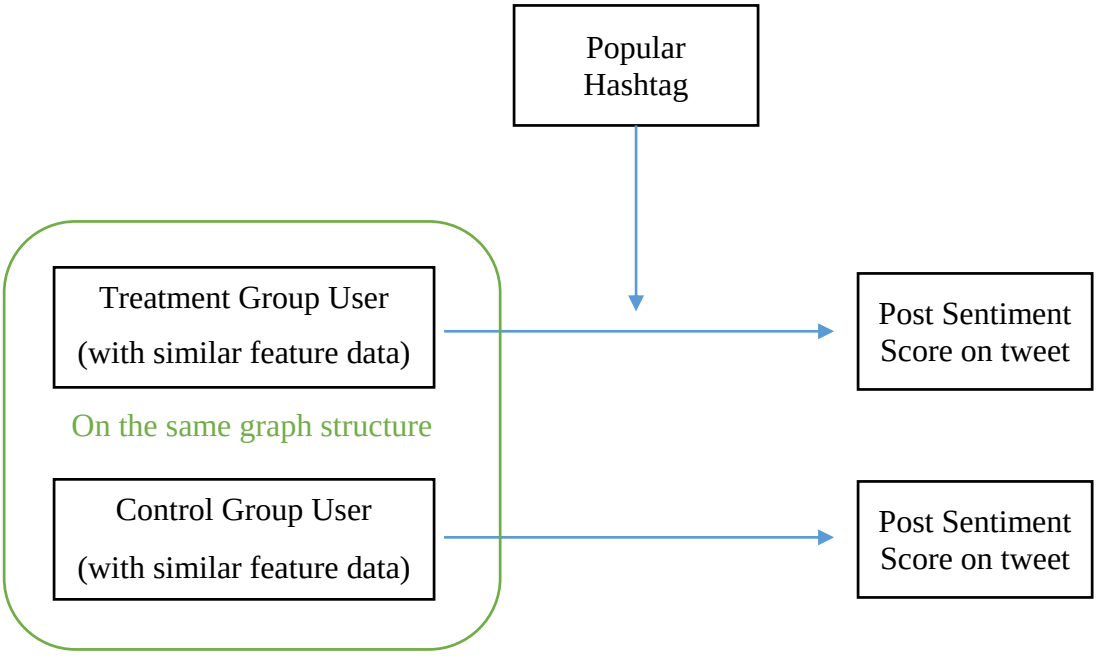
The organization of the model building section is as follows: We first demonstrate the research context and summarize the general framework in this study. Second, we review the Propensity Score Matching Method (PSM) based on the Rubin's Framework, and discuss the challenges of PSM in our Twitter context. Third, we discuss how the representation learning method can help address the challenges PSM faces. Fourth, we discuss how to use Graph Neural Network to address the challenge of large-scale social network. embedding method into a deep neural network framework. We also want to note that while there is a significant body of literature on causal inference using causal graph (Pearl 2009), our framework focus on counterfactual inferences first proposed by Rosenbaum & Rubin (1983).

3.1. Summary

In this section, we first briefly introduce the research context. We then briefly summarize the deep learning framework employed in this research.

Our research application is as follows, we can observe the treated group user u and the control group user v on twitter. Both of them can receive some tweets on twitter. These tweets includes some popular hashtags and we use them as the treatment t. These users post some comments on these tweets and is marked with the sentiment scores. We use these sentiment scores as outcome y. We want to test how the popular hashtag influences on the sentiment scores.

Figure 1. Concept Model



We use the graph representation learning method to calculate the treatment effect. Note that any user $i \in \{1,2,\ldots,n\}$ is a user on a social network platform. If the user i retweets the prior tweet given by user j, it is defined as a social tie between each users. We define the user as node, the soical ties between each two users as edge. Then we can construct the retweets graph G based on what we can observe. Therefore, for each user, we can observe his or her demographic data, (user location, followers count, friends count, etc) and we denote it as x_i, the covariate variables. He or she leaves comments containing some sentiment score (number from -1 to 1) and we denote the sentiment of tweet m by user i as y_{im}, the outcome. If we use k to represent the topic[2] of this tweet, we assume that the sentiment of this tweet is influenced by other tweets with the same topic q that user i retweeted before she posts tweet m. Thus we also use t_{im} to represent whether user i is exposed to other tweets belong to the same topic q before she wrote tweet m.

[2] We operationalize topics of tweets using hashtags (such as #Narendra Modi, #BJP, etc). We can also use NLP to identify tweets topics as well.

This t_{im} is a binary variable and the treatment we consider in this context. The retweets graph G can be transformed into an adjacency matrix A to represent the edges and nodes.

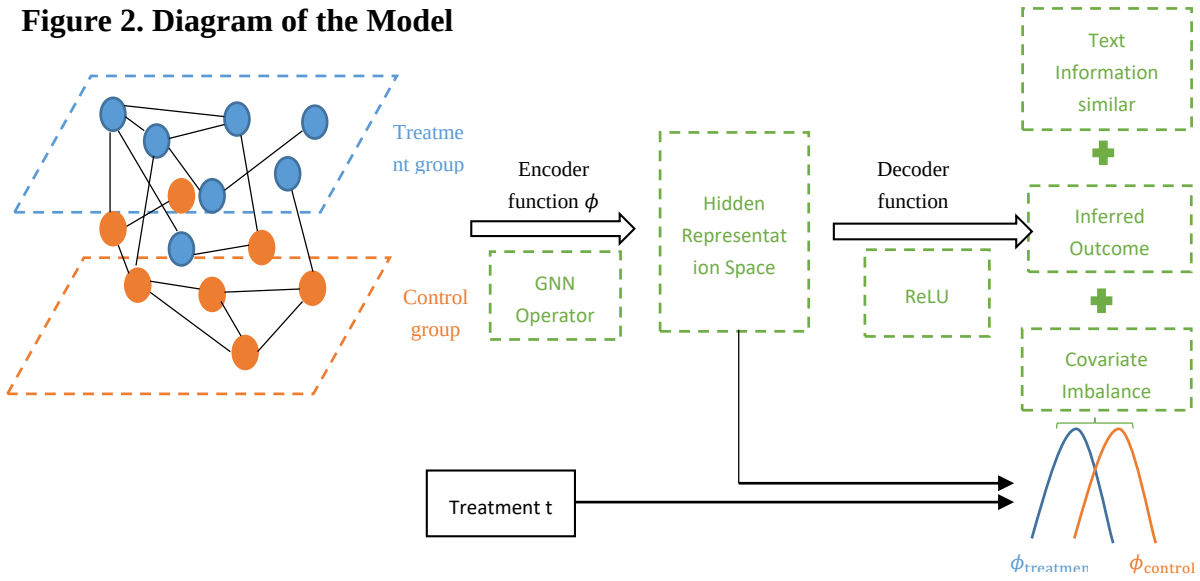
For each observation, we can get a tuple $(\{x_i, y_{im}, t_{im}\}, A)$. Note that the adjacency matrix A is huge for the social network embedded on Twitter. Thus we use Graph Neural Network (GNN) to generate representation space (termed as hidden embedding space in GNN literature) of the original social graph, thus reduce the dimension of original social network. We use encoder function ϕ in figure below to represent this process. Then we use the decoder function ψ to reconstruct each node $\psi(\phi(x, A), T)$ into the original sample space to construct the labels for counterfactual sample set. We further find the match between factual and counter factual sample set by minimizing the distance of these two sample sets in representation space. Then we can calculate the

Individual Treatment Effect (ITE) as:

$$ITE_{im} = \big(y_{im}(1) - y_{im}(0)\big) \mid X, A$$

Where the $y_{im}(1)$ means the units receives the hashtag, and $y_{im}(0)$ means the units do not receive the hashtag. We will discuss these steps in more details in sections below.

Figure 2. Diagram of the Model



3.2. Counterfactual Inferences and Propensity Score Matching

In this section, we first review counterfactual inferences framework first developed by Rosenbaum & Rubin (1983), and discuss the challenges we face when adapting the counterfactual inferences framework in our online social media context. We further discuss the popular propensity score matching (PSM) method under Rosenbaum & Rubin's framework, and how PSM is a special case of the representation learning method.

There are n observations notated as $\{X, Y, T\}$,where X means covariates (features, i.e. demographics information), Y means outcome (i.e. tweets sentiment), and T means treatment (i.e. whether user i is exposed to others' tweets of the same topic before she posts m^{th} tweet). For the m^{th} tweet by user i, we can get the data tuple for each user as $\{x_i, y_{im}, t_{im}\}$. Treatment can only choose 0 or 1.[3] Therefore, we denote $y_{im}(0)$ as control group outcome and $y_{im}(1)$ as treated group outcome. We want to highlight that following Rosenbaum & Rubin's framework, we also need to construct "counterfactual outcomes" both control group outcome and treated group outcome. For example, if we actually observe before user i posts the m^{th} tweet, she is exposed to a tweet with same topic, this is called "factual outcome". Thus the counterfactual outcome is the one that does not happen in reality, i.e. she was not exposed to any tweet with same topic before she posts the m^{th} tweet. We define $y_{im}(1)|t_{im} = 1$ and $y_{im}(0)|t_{im} = 0$ as factual outcome and $y_{im}(1)|t_{im} = 0$ (i.e. what would happen for a user in treated group if she were receiving no treatment) and $y_{im}(0)|t_{im} = 1$ (i.e. what would happen for a user in control group if she were receiving treatment) as counterfactual outcome. We will discuss in more details on the construction of counterfactual outcomes in following sections.

[3] We can easily extend our model to a multiple treatment scenario in which the treatment will represent the number of tweets a user is exposed to.

We want to note that one of the important assumptions under Rosenbaum & Rubin (1983) Framework is *unconfoundedness*. Specifically, there is no hidden confoundedness effect on both X, Y, and T:

$$y_{im}(0), y_{im}(1) \perp t_{im} \,|x_i$$

And we will talk about how our method can relax this assumption in the following sections.

The Individual Treatment Effect (ITE) can be defined as:

$$ITE_{im} = y_{im}(1) - y_{im}(0) \tag{1}$$

The Average Treatment Effect (ATE) can be defined as:

$$ATE = E[Y(1) - Y(0)|X] \tag{2}$$

The golden rule of causal inference is a randomized experiment in which treated and control group are assigned randomly. In real world, to mimic the random assignment of treated group and control group, many scholars use propensity score matching method to calculate the casual effect (Imai et al 2014, King and Nielsen 2019). However, traditional PSM suffers from two problems.

First, PSM requires us to transform the covariate variables X into the propensity score by a logistic function: $\pi_i = Prob(t_i = 1|X) = \frac{1}{1+e^{X_i\beta}}$. PSM works well if the covariate variables sample space is very small and the distributions of the covariate variables has no significant differences between treated and control groups. In this case, when PSM generates propensity score and match treated and control group units based on the distance of between treated and control group units, units with similar propensity score are also likely to be similar pairs (similar location) in covariate variables sample space :$Distance = |\pi_{treated} - \pi_{control}|$. In other words, PSM can be considered as a one-dimensional representation space of the covariate variables

sample space, and PSM works well only if there is relatively little information loss compared with the covariate variable sample space.

However, social networks on social media platforms like Twitter often have very complex network structures. If we incorporate these complex network structures as the covariate variables in PSM, it leads to a very large covariate variables sample space with significantly different distributions of the covariate variables between treated and control group. As such, the one-dimensional representation space constructed by PSM is unlikely to represent all these covariate variables very well (Li and Fu 2017). Two users may have big difference on covariate variables but small difference on the propensity scores. This leads to include the wrong information problem (Imai et al 2014, King and Nielsen 2019).

Second, the *unconfoundedness* assumption is critical for the performance of PSM. In other words, PSM works well if there is no other unobserved confounding factors that change the relationship between X and Y. Yet this assumption is easily violated in our Twitter context such as homophily—a user posts a tweet favoring Democrat following another positive tweet about Democrat from her friend not because her friend's tweet has influence on her, but rather because both of them share similar political opinions. Therefore, such confounding factors will lead the propensity score matching method to calculate the bias result. In other words, we cannot construct the balanced propensity score when there exists the confounder (Imai and Ratkovic 2014, King and Nielsen 2019, Sant'Anna et al 2020). Please also refer to appendix 3 for more detailed discussions.

Also note that, the matching process in PSM can be considered as the process of constructing the counterfactual outcomes described in Rosenbaum & Rubin (1983), because the matched

observations in control groups can be regarded as the counterfactual of corresponding observations in treated group.

Overall, the logistic regression transforms the covariates variables into the propensity score space. Then we use these propensity score to represent the observations. The treatment effect is only calculated by these covariates variables. This calculation is efficient under the balanced covariates variables situation. Therefore, we can treat the propensity score matching method as a special case of the representation learning method. It works well under the small balanced covariates variables space and linear relationship between covariates and the outcome.

In the next subsection, we will describe the representation learning framework we adapt in this research, and discuss how it can address the two problems PSM faces in our social media context.

3.3. Representation Learning for counterfactual framework

If we can observe both factual outcomes and counterfactual outcomes, it becomes a supervised learning problem. However, we cannot observe the counterfactual outcomes, i.e. $y_{im}(1)|t_{im} = 0$ (treated group if there were no treatment) and $y_{im}(0)|t_{im} = 1$ (control group if there were treatment) in previous section. In this study, we use the representation learning method described below to construct these counterfactual outcomes. (Please check the appendix 4)

Let‘s take the example as shown in table 1. In the factual sample set, let’s assume the observed covariates are tweets language and location. We also observe whether focal user receives treatment or not (whether focal user is exposed to others’ tweets with same topic, such as #BLP, etc), as well as the corresponding outcome (tweets sentiment). In the counterfactual sample set, we need to reverse the treatment variable value, so that in the first row of Table 1, the value of Treatment variable is 1 in factual sample set (marked in red), and the value of Treatment variable

is 0 in counterfactual sample set (marked in green). If we compare the factual sample set and the counterfactual sample set, their covariate variables, X's, are the same, yet we only observe the outcome Y for factual sample set. To construct the outcome Y for counterfactual sample set, we employ Domain Adaptation (Cortes and Mohri 2014, Long et al 2014), where we want to label the outcome of counterfactual sample set using training results based on the factual data set in which we do have the outcome value. To some extent, this process is similar to finding the best matched units in factual sample set for units in counterfactual sample set.

Table 5. An example of the factual sample set and the counterfactual sample set

Covariates: X = (Location , Language) , Treatment: t={0,1}			
Factual sample set		Counterfactual sample set	
X: (Location, Language, Treatment)	Y: Sentiment score after treatment	X: (Location, Language, Treatment)	Y: Sentiment score after treatment
(Delhi, English, 1)	$Y_1 = 0.1$	(Delhi, English, 0)	$Y_1 = ?$
(Bangalore, English, 1)	$Y_1 = 0.3$	(Bangalore, English, 0)	$Y_1 = ?$
(Bangalore, Hindi, 0)	$Y_0 = 0.4$	(Bangalore, Hindi, 1)	$Y_0 = ?$
(Bangalore, Hindi, 0)	$Y_0 = -0.2$	(Bangalore, Hindi, 1)	$Y_0 = ?$
(Bangalore, Bengali, 0)	$Y_0 = -0.7$	(Bangalore, Bengali, 1)	$Y_0 = ?$

Step 1. Construct representation space

The first step of constructing counterfactual sample set is to transformation observed covariates to the representation space. Observed covariates sample space in social media study is often highly dimensional as it involves complex social network structure. To reduce the dimension of covariates sample space, PSM generate propensity scores which can be considered as a one-dimensional representation space of the covariate variables. Remember in previous section, we argue that the first problem of PSM is that PSM lose a lot of information in the process of

calculating propensity scores. To overcome this challenge, in this study, we use the *encoder function* ϕ to map covariates into the representation space, $\phi(\cdot)$. For covariates X of an observation, we denote the corresponding vector $\phi(X)$ as the corresponding ***representation***. Notice that the use of representation space allows us to lose much less information because of two reasons. First, the constructed of representation space is allowed to be multi-dimensional, in comparison with one-dimensional representation space based on propensity score. Second, we can also minimize the information loss by using a deep learning structure with multiple layers of non-linear functions. In comparison with the logistic model that PSM uses to calculate propensity score, the representation space generated using a deep learning structure can help us keep more information from the original sample space (Li and Fu 2017, Johansson et al 2020). The inclusion of more information is helpful for finding matching units from factual and counterfactual sample sets, and consequently construct labels for counterfactual sample set.

Step 2. Distance in representation space

In step 2, we minimize the distance between the representations of factual sample set and the representations of counterfactual sample set in the representation space $\phi(X)$. Because of the use of a multi-dimensional representation space, we can now use the Maximize Median Distance or Wasserstein Distance to measure the distance between two sets in the representation space, denoted as $Distance\ (X_{Factual}, X_{CounterFactual})$. The use of these two types distances allows us to retain more information compared with the distance used in PSM. Thus matching based on these two distances can help us find better matching in the original covariates sample space. We provide more details of these two distance measures in Appendix 5 and 6.

Step 3. Use factual set to learn the label of counterfactual set

Keep in mind, if we were to estimate the causal effect, we still need to know the label (i.e. outcome Y) for counterfactual sample set. We use *decoder function* $\psi(\cdot)$ as the learned causal relationship among representation space $\phi(X)$, treatment *T* and outcome *Y*. We use ReLU as the activation function in $\psi(\cdot)$ in this study. Then together with the distance function in step 2, we can define the loss function as:

$$L = L_{MSE}(\psi(\phi(X), T), Y) + L_{Distance}\left(\phi(X_{Factual}), \phi(X_{CounterFactual})\right) + \lambda * R(h) \quad (3)$$

By minimizing this loss function above using our deep learning framework, we can learn both encoder function $\phi(\cdot)$ and decoder function $\psi(\cdot)$. Note that the first term $L_{MSE}(\psi(\phi(X), T), Y)$ is the error between $\psi(\phi(X), T)$ and Y. The second term $L_{Distance}(\phi(X_{Factual}), \phi(X_{CounterFactual}))$ is the distance of representation learning in step 2. The third term $\lambda * R(h)$ is a regularization term that can be used to capture other similarities between factual and counterfactual units. After we learn $\phi(\cdot)$ and $\psi(\cdot)$, we can then calculate the labels of counterfactual sample set.

Except calculating the distance between the factual group and the counterfactual group in terms of the demographic information, we also include the text information into the loss function. By adding the distance between the factual group and the counterfactual group in terms of the text information, we can get

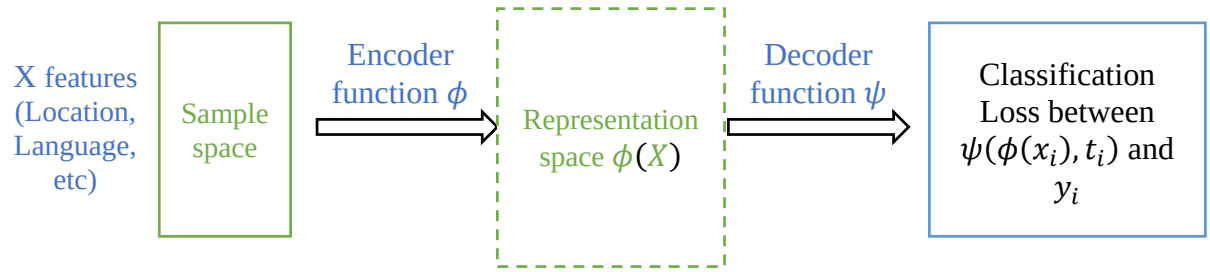
$$L = L_{MSE}(\psi(\phi(X), T), Y) + L_{Distance}\left(\phi(X_{Factual}), \phi(X_{CounterFactual})\right) + L_{Distance}(topics(X_{Factual}), topics(X_{Counterfactual})) + \lambda * R(h) \quad (4)$$

In addition to calculating the label of counterfactual sample set, inclusion of decoder function can also help use learn the nonlinear relationship among covariates, treatment and outcome, especially in the presence of social network data. While an important assumption in Rosenbaum

& Rubin (1983) framework is the unconfoundedness (i.e. there is no confounder C to affect the relationship between X, Y ,and T), this assumption could be difficult to satisfy in our social network context due to confounding factors such as homophily. In this study, by using a deep learning method to infer the nonlinear causal relationship $\psi(\cdot)$ among covariates, treatment and outcome, we can significantly reduce the bias of estimated treatment effect.

Formally, the model building of the representation learning is as follows: we define a hidden space $\mathcal{H}$ and two functions ϕ and ψ. The first function ϕ is for mapping X into $\mathcal{H}$ and the second function ψ is for mapping $\mathcal{H} \times$ T into Y. Then by minimizing the loss function above, we should learn the appropriate representation functions ϕ and ψ to get the label of counterfactual sample set: $\hat{Y} = \psi(\phi(X), T)$.

Figure 3. Diagram of the Counterfactual Inference-Step 1

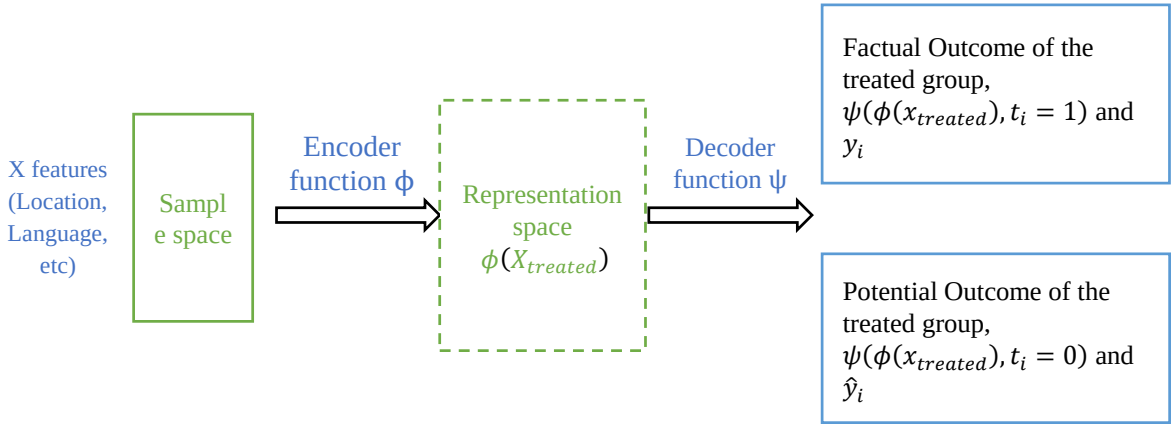


In order to operationalize the three steps above efficiently, we use the method suggested by Johansson et al (2016). Note that when we minimize the loss function in equation (3), this will also produce a match of factual and counterfactual units.[4] In the meanwhile, after we learn the appropriate representation function $\psi(\cdot)$ and $\phi(\cdot)$, we can use the $\psi(\phi(X_{treated}), T = 0)$ to calculate the label for each unit in counterfactual sample set simply by changing the value of

[4] The treated group users with treatment, $y_i(1)|t_i = 1$ (this is in factual sample set) are matched with the treated group users without the treatment (this is in counterfactual sample set), $y_i(1)|t_i = 0$. Same way, the control group users without treatment, $y_i(0)|t_i = 0$ are matched with the control group users with the treatment, $y_i(0)|t_i = 1$.

treatment T, denoted as $\hat{y}_{im}(1)|t_{im} = 0$. Similarly, we can use the $\psi(\phi(X_{control}), T = 1)$ to calculate the label for each unit in counterfactual sample set, denoted as $\hat{y}_{im}(0)|t_{im} = 1$.

Figure 4. Diagram of the Counterfactual Inference-Step 2



Thus, we can write the individual treatment effect as:

$$\widehat{ITE}_{im} = \begin{cases} (y_{im}(1)|t_{im} = 1) - (\hat{y}_{im}(1)|t_{im} = 0), if\ t_{im} = 1 \\ (\hat{y}_{im}(0)|t_{im} = 1) - (y_{im}(0)|t_{im} = 0), if\ t_{im} = 0 \end{cases}$$

$$= \begin{cases} (y_{im}(1)|t_{im} = 1) - \psi(\phi(x_{im}), t_{im} = 0), if\ t_{im} = 1 \\ (\psi(\phi(x_{im}), t_{im} = 1) - (y_{im}(0)|t_{im} = 0), if\ t_{im} = 0 \end{cases} \quad (5)$$

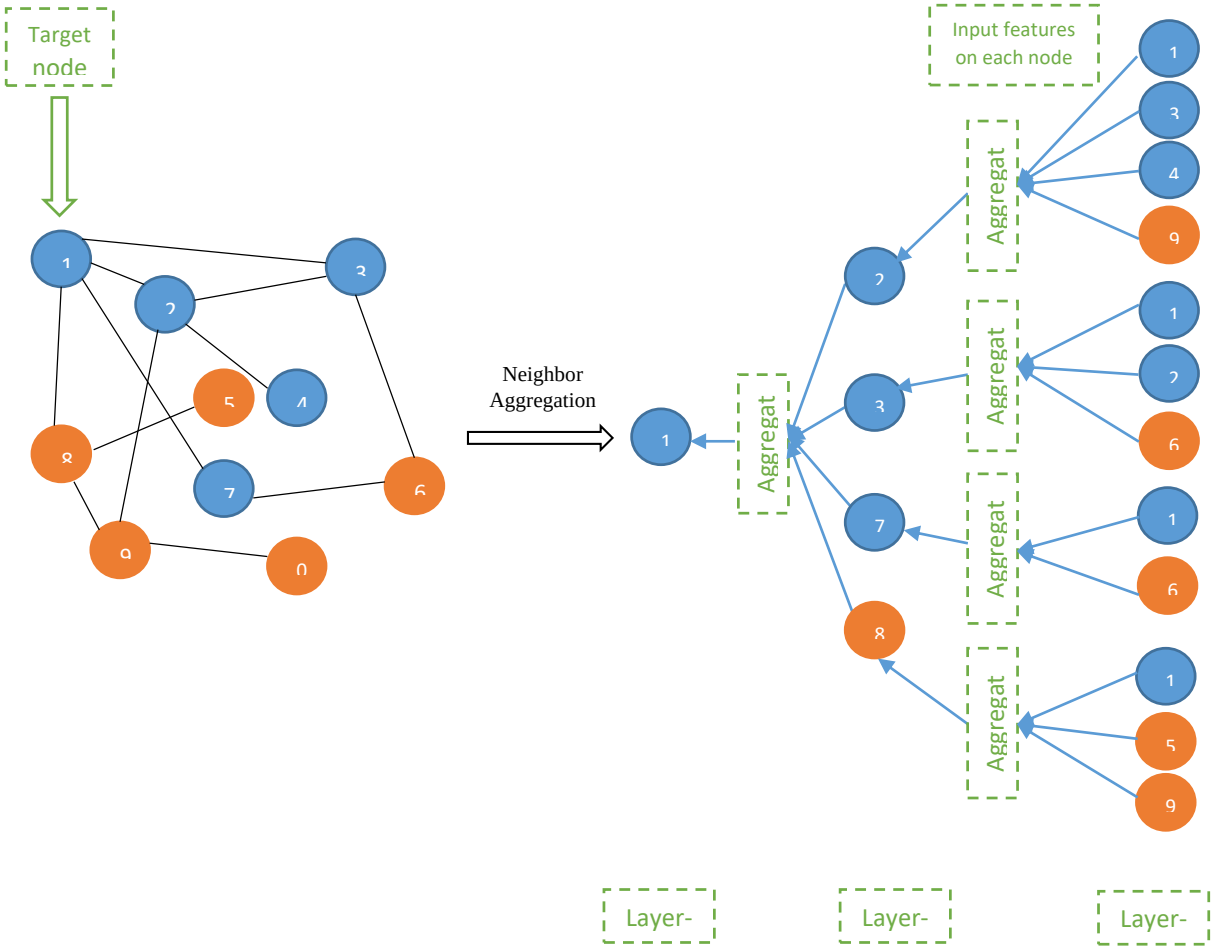
3.4. Deep Graph Neural Network.

We describe the causal inference method we use in section 3.3 where we minimize the loss function in equation (3). In this section, we discuss in more detail how to obtain the encoder function $\phi(\cdot)$ in equation (3).

Individual position within the social network on Twitter plays an important role in predicting her tweet sentiment. Yet one of the challenges in leveraging social network information on Twitter is

network's large-scale. Previous literature using Twitter network information mainly rely on structured network characteristics such as degree centrality, tie strength, structural hole (Bonacich 1987, Borgatti 2005, Krivitsky et al 2009, Song et al 2019, Rishika and Ramaprasad 2019, Mousavi and Gu 2019). While these network characteristics can help address the large-scale problem in Twitter related study, which network characteristics to use often requires context-specific feature engineering. In our study, by using a Graph Neural Network (GNN) structure, we can generate a representation (also called *embeddings*) of networks on Twitter which reduces the dimension of network structure while keeping most of the network information. This allows us to solve the challenge of large-scale network. As such, context-specific feature engineering is no longer needed. In addition, information loss due to feature engineering can also be avoided. This is because it is difficult to include all available network characteristics into a traditional feature-engineering machine learning algorithm. And some network characteristics potentially for feature engieenring, such as betweeness centrality, are very computationally expensive to calculate for large-scale networks. Note that GNN is a *deep* node embedding technique, an improvement over the *shallow* node embedding techniques. Please refer to Appendix G for the details of one shallow node embedding technique.

Figure 5. Diagram of the Graph Neural Network



In our GNN framework, the network can be described using tuple $\{V, E, X\}$ dataset, where V represents vortexes, E represents edges and X are the node features such as location and language of tweets. For a network with one million nodes, the corresponding adjacency matrix will be one million by one million. To reduce the dimension of this huge matrix, we use GNN so that we can generate a representation (also called an *embedding*) for each node with much smaller dimension (for example, a vector with size 30 for each node), yet keep most of the network information associated with focal node. The intuition of GNN is that we aggregate the neighbor information of each focal node using neural network method. Note that the neighbor here not only include the

immediate neighbor of focal node, but also include neighbors that are connected through multiple edges.

Consider node 1 in figure above. Node 1's immediate neighbors are nodes 2,3,7,8. Node 2 itself also has neighbors 1,3,4,9. Consider the simple case in which we only include neighbors that are at most two edges away, to calculate the embedding of node 1. To calculate the embedding of node 1, we will need to aggregate information of node 2,3,7,8. This requires us to calculate the embedding of node 2,3,7,8 respectively (Layer-1 in figure above) before we calculate the embedding of node 1. As such, we will first calculate node 2 embedding using embeddings of node 1,3,4,9 (Layer-0 in figure above), node 3 embedding using embeddings of node 1,2,6 and etc. Then we can calculate the embedding of node 1 using embeddings of node 2,3,7,8. More formally, we use h as hidden embeddings of the neighbors. The number of layers of hidden embedding h is denoted as K, and K can be any value (in this example, $K = 2$). Note that embeddings in Layer-0 are one-hot enbeddings of node features. The final layer of hidden embedding (Layer K) will be the resulting embedding of focal node that we want to calculate. Formally, we use $\mathrm{h}_v^{\mathrm{k}}$ to represent the hidden embedding of a node v in Layer $k \in \{1,2,\dots,K\}$.[5] We will iteratively update this hidden embedding $\mathrm{h}_v^{\mathrm{k}}$ by incorporating the aggregated information from its immediate neighbors, as well as the information of the node itself in previous iteration. We use $\mathrm{m}_v^{\mathrm{k}}$ to represent this aggregated information from immediate neighbors of this node. In other words, we can write the l^{th} iteration in this updating process as:

$$(\mathrm{h}_v^{\mathrm{k}})^l = update\left((\mathrm{h}_v^{\mathrm{k}})^{l-1}, (\mathrm{m}_v^{\mathrm{k}})^{l-1}\right) = update\left(W_{self}^T(\mathrm{h}_v^{\mathrm{k}})^{l-1} + W_{Neighbor}^T(\mathrm{m}_v^{\mathrm{k}})^{l-1}\right) \quad (5)$$

[5] For example, to calculate embedding of node 1 in previous example, we will need to aggregate information of node 2,3,7,8 in Layer-1. In other words, we need to calculate h_2^1, h_3^1, h_7^1, h_8^1.

Here we use sigmoid function as the update function. After training the model by L^{th} iteration, we get the final state z to represent the final state representation: $z_v = h_v^L$. This final state representation z_v is just the $\phi(v)$ of the encoder function in section 3.3.

To calculate m_v^k in equation (5), we aggregate information from node v's immediate neighbor following the aggregate equation (6) below. The number of immediate neighbor nodes of node v is $\mathcal{N}$. $\mu \in \mathcal{N}(v)$ is an immediate neighbor of node v. W_k and B_k are the weight and bias of neural networks respectively for this layer k that we need to estimate. We put all them together into a ReLU function $\sigma(\cdot)$. Note that the hidden embedding of node v in layer k depends on the hidden embedding of v's immediate neighbor in previous layer $h_\mu^{(k-1)}$.

$$h_v^k = \sigma(W_k \sum_{\mu \in \mathcal{N}(v)} \frac{h_\mu^{(k-1)}}{|\mathcal{N}(v)|} + B_k h_v^{(k-1)}) \qquad (6)$$

We also want to note that the three popular GNN methods, GCN (Kipf T. N. and Welling M. 2017), GAT (Velickovic et al 2018) and GraphSage (Hamilton et al 2017), are variations of GNN with different aggregate functions and update functions.

4. DATA AND RESULT

4.1. Data Description: Indian General Election Twitter 2014

Our dataset is the Indian General Election data from the Twitter between Jan 1st and May 16th 2014. This dataset contains all tweets related to Indian General Election. This dataset contains 3.2 millions tweets, from 218, 7334 users. After data cleaning, we can get the tweets sentiment, *Y*, the treatment *T*, (i.e. whether user i is exposed to others' tweets of the same topic before she posts m^{th} tweet), the covariates variables *X* such as User Location, User Followers Count, etc, and the adjacency matrix *A*.

The tweets sentiment *Y* is a numerical number from -1 to 1. Negative/Positive numbers reflect users' negative/positive attitudes within the tweets.

The treatment *T* is the binary number to represent whether a user i receives the treatment, i.e. whether users were exposed to other tweets with the same topic. In this research, we use hashtag, such as #Narendra Modi, #BJP, #AAP, etc., to categorize the topics of tweets. Note that we only include hashtags that represent political preference information in this study.

These covariates variables *X* include two types of data, network theory-based variables, and demographical variables. The network theory-based variables means the features we can observe from the social network. It reflects how users interact with their neighbors, how their neighbors affect the given users, and etc. These network theory-based variables include: 1) user followers count, which means the number of followers of the focal user; 2) friends count, which means the number of users with whom the focal user follow each other; 3) listed count, which means the number of users who added focal user to a list.

The demographical variables include: 1) user location, which reflects where does users post the tweets; 2) status language, which language does user post on twitter; 3) status source, which channel does the user post the tweets (e.g. web, iphone and etc.).

We also construct an adjacency matrix $A_{\tilde{t}}$, which is a large sparse matrix to represent the retweets network between each users on twitter before period $\tilde{t}$. The value of element $A_{ij\tilde{t}}$ of this matrix represent whether user i retweeted a tweets from user j before period $\tilde{t}$. While previous literature uses network-based characteristics as covariates in estimation model, in this study, we use GNN to help us extract graph embedding features. We argue that these graph embedding features can significantly improve the predictive performance as it allows us to capture more complicated network features that are otherwise omitted by network theory based characteristics.

Table 6. Data description

Variable	Description
User Location	Delhi, Bangalore, etc. We decode it as the category value
User Followers Count	Number of followers for the user
User Friends Count	Number of friends for the user
User Listed Count	Number of listed members
Status Language	English, Hindi, Bengali, etc. We decode it as the category value
Status Source	Web, Iphone, and Roundteam. We decode it as the category value
Status Content	Text
Status Time	Day time
Hashtag	#Narendra Modi For PM, #BJP, #ANI, etc
Sentiment Score	Number (From -1 to 1)

4.2. Semi-Synthetic Data

In order to evaluate the performance of the model, we need to compare our model with other traditional methods based on the differences between estimated results and ground truth results.

Following the previous literatures (Johansson 2016, Louizos 2017, Veitch et al 2019), we need to construct the semi-synthetic data based on what we can observed from the real world. We need to construct the factual set includes the treated group with the treatment and the control group without the treatment and the counterfactual set includes the control group with the treatment and the treated group without the treatment. We denote $y_{im}(0)$ as control group outcome and $y_{im}(1)$ as treated group outcome. Previous literature also suggests that the sentiment of focal user's tweet is influenced by both neighbors' demographic features as well as the content of their tweets (Veitch et al 2019, Ma et al 2020, Guo et al 2020)

Specifically, we follow the steps below to construct the semi-synthetic dataset.

In step 1, we calculate the distribution of sentiments for user i, denoted as $sentiment_{i,k}$, if this focal user is exposed to tweets with same topic k before. Then, we calculate the sentiment distribution in the treated group and the control group. The $sentiment_k^{treated}$ denotes the sentiment distribution in the treated group if he or she is exposed to tweets with the same topic k before. The $sentiment_k^{control}$ denotes the sentiment distribution in the control group if he or she is exposed to tweets with the same topic k before.

In step 2, we calculate the sentiment distribution of focal user i's neighbors j . We use $\sum_{j\in N(i)} sentiment_{j,k}^{treated}$ and $\sum_{j\in N(i)} sentiment_{j,k}^{control}$ to represent the sentiment distribution of focal user's neighbors in the treated group and the control group respectively.

Then, we denote the affection from user's neighbor's opinions, as a_{im}^0 and a_{im}^1. Intuitively, $\boldsymbol{a_{im}^0}$ and $\boldsymbol{a_{im}^1}$ are two vectors representing the probability that the m^{th} tweet by user i belongs to control group and treatment group respectively. These two vectors, $a_{im,k}^0$ and $a_{im,k}^1$, are the probability that the corresponding tweet belongs to topic k.

$$a_{im,k}^{1} = sentiment_{i,k} * \left(sentiment_{k}^{treated}\right)^{c} + \sum_{j \in N(i)} sentiment_{j}^{treated} * \left(sentiment_{k}^{treated}\right)^{c}$$

$$a_{im,k}^{0} = entiment_{i,k} * \left(sentiment_{k}^{control}\right)^{c} + \sum_{j \in N(i)} sentiment_{j}^{control} * \left(sentiment_{k}^{control}\right)^{c}$$

where $N(i)$ is the set of focal user i's neighbors, $\left(sentiment_{k}^{treated}\right)^{c}$ means the centroid of sentiment distribution on treated group, and $\left(sentiment_{k}^{control}\right)^{c}$ means the centroid of sentiment distribution on control group.

Step 3, We calculate the outcomes of the treated group and the control group as follows:

$$y_i(t_i) = (1 - t_i)a_i^0 + t_i a_i^1 + \epsilon, \qquad where\ \ \epsilon \sim N(0,1)$$

Where the treatment t_i follows the binomial distribution. If $t_i = 1$, the outcome of the treated group $y_i(1) = t_i a_i^1 + \epsilon$. If $t_i = 0$, the outcome of the control group $y_i(0) = a_i^0 + \epsilon$.

Combining both the treated group outcome and the control group outcome, we can create the semi-synthetic dataset for estimation.

Recall that we have Average Treatment Effect as:

$$\tau_{ATE} = E[Y(1) - Y(0)|X, A] = \frac{1}{n}\sum_{i=1}^{n}[Y_i(1) - Y_i(0)|X, A]$$

The outcome $Y(1)$ and $Y(0)$ is what we observe from data

The estimated Average Treatment Effect is:

$$\hat{\tau}_{ATE} = E\left[\hat{Y}(1) - \hat{Y}(0)\middle|X, A\right] = \frac{1}{n}\sum_{i=1}^{n}\left[\hat{Y}_i(1) - \hat{Y}_i(0)\middle|X, A\right] \tag{11}$$

The estimated outcome $\hat{Y}(1)$ and $\hat{Y}(0)$ is calculated by the graph neural network.

Normally, we choose the **mean absolute error (MAE)** as the evaluation metrics:

$$\epsilon_{MAE} = |\tau_{ATE} - \hat{\tau}_{ATE}| \tag{12}$$

Another common evaluation metrics is **mean squared error (RMSE)**:

$$\epsilon_{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tau_{ATE} - \hat{\tau}_{ATE})^2} \quad (13)$$

4.3. Model Performance

As we described in more details in section 3, compared with traditional method, our model makes two improvements on the construction of balanced dataset. First of all, we improve the balance of the dataset because we now incorporate complicated network features generated from GNN that are otherwise omitted by traditional network theory based covariates. Second, we use the representation learning method to match observations in treated and control groups in a high dimensional space. This is in comparison with traditional methods such as PSM in which only a one-dimensional statistic (i.e. propensity score) is used to match observations in treated and control group. We employ analysis in section 4.3.1 below to demonstrate how GNN can improve the balance of the dataset, and use section 4.3.2 to demonstrate how representation learning improves the estimation of the treatment effect.

4.3.1. Covariates Balanced Improvements from GNN

As we discussed in section 3, the balance of treated and control group observations critically depends on covariate variables. Following the prior studies (Ho et al 2007, Austin and Peter 2009, Heller et al 2010, Ali et al 2014, Imai et al 2014), we conduct the balanced test and match observations in treated and control group using network theory based features only (the balanced test is based on the nearest neighbor matching method). We then conduct the balanced test and match observations in treated and control group by further incorporating network embeddings generated from GNN. We demonstrate covariates balanced improvement by randomly sample 10000, 20000,40000, 80000, 100000 users. We provide the details of these results in Appendix 8. We observe that by incorporating network embeddings, we can generally improve the balance of the dataset. In particular, we present the balance test results based on propensity score from

100000 users sample in Table 7 below. We also present the balanced test result for the original imbalanced dataset in this table. We focus on three measures of balanced test. First, the Standardized Mean Differences (Std. Mean Diff.), measures the differences between the mean values on two groups. Second, the Variation Ratio (Var. Ratio), measures the differences between the variance values on two groups. Third, the Mean Differences Empirical Cumulative Density Function (eCDF) measures the differences between the cumulative distributions on two groups. Criteria: If two groups (the treated group and the control group) with Standard Mean Difference close to zero, with Difference Variation Ratio close to one, and with eCDF Mean Difference value close to zero is considered to be balanced.

First, dataset with matching based on network theory covariates and that with matching using GNN network embeddings can significantly improve the balance of the original imbalanced dataset. More importantly, by incorporating GNN generated network embeddings, we can further improve the balance of the dataset along all three dimensions (i.e. Standard Mean Difference, Difference Variation Raio and eCDF Mean Difference). The details of the balanced test for each network theory based features and GNN embedding features, such as datasets with difference user sample size and balanced results for each covariates, are available in Appendix 8.

From Table A.3 to Table A.6. As we can see from Table A.3 , the balanced covariates measures mean the treated group and the control group has no significant difference on each covariate features. We list a balanced covariates features (without graph embedding features) comparison on the appendix Table A.3. There is no big difference on mean value and standard error. Also, we list an imbalanced covariates features (without graph embedding features) comparison on appendix table A.4. There is a big difference on mean value and standard error. We will show

that the deep graph representation learning is a robust method under both balanced covariates situation and imbalanced covariates situation.

For table A.5 and table A.6, we extract some graph embedding features by using GNN method. Therefore, when we compare the covariates features on the treated group and the control group, there are more graph embedding features that can be used. The table A.5 shows there is no big difference on the mean value and the standard error value. The table A.6, shows there is a big difference on the mean value and standard error value.

The details of the propensity score balanced improvement on different scale of the network are shown from appendix Table A.7 to Table A.11. In most cases, after we conduct the balanced test on the dataset with GNN embedding features can always contributes a better performance. It demonstrate that our method can improve the balance of the dataset.

From Table A.7 and A.8, we show how network theory based features distribute on the treated group and the control group. From Table A.9, A.10 and A-8.11 we show how network theory based features and the graph embedding features distribute on the treated group and the control group.

We summary the results from the Table A.7 to Table A.11 into the Table 7. As we discussed in section 3.2, the propensity score is a representation of the covariate variables. Therefore, we compare four different datasets with their propensity scores and check how the balanced got improvement. The Table 7 shows that the dataset including the GNN features, after the nearest neighbor matching, give the most expected balanced performance compared with three other cases.

Table 7. Propensity Score Balanced Improvement (100k users)

	Std. Mean Diff	Diff. Var. Ratio	eCDF Mean Diff
Original Imbalanced Dataset	0.3845	0.3994	0.0658
After Nearest Neighbor Matching	0.0004	1.0022	0.0001
Imbalanced Dataset with GNN	0.5915	0.6630	0.1451
Nearest Neighbor Matching with GNN	**0.0002**	**1.0009**	**0.0000**

4.3.2. Treatment Effect Improvements from Representation Learning

To demonstrate the performance of our proposed model, analyses in this section are based on a semi-synthetic data as it provides a ground truth in which we can compare with. Results suggest that the GNN embedding can significantly improve the data balance, and Representation Learning can help better identify the treatment effect compared with benchmark methods.

To demonstrate the performance improvement using representation learning in our method, we replace the representation learning component of our model with three benchmark models. This allows us to inspect how much representation learning can improve the balance of the resulting data. We list three traditional methods which we use to replace representation learning:

Propensity Score Matching: this method calculates the propensity scores of observation i in treatment group and j in control group. We can then calculate treatment effect after we find matching pairs in terms of propensity score using formula below:

$$\hat{\tau}_{ATE}^{PS} = \left[\sum_{i:t_i=1} (y_i - y_j) - \sum_{i:t_i=0} (y_j - y_i) \right] / n$$

Inverse Probability Treatment Weight: After we calculate the propensity scores, we can further calculate the probability of receiving treatment of each observation, and use this

probability as the weight to construct a randomly controlled trial setting. This weight is calculated using formula below:

$$\mathrm{w_i} = \frac{t_i}{P(t_i|x_i)} + \frac{1-t_i}{1-P(t_i|x_i)}$$

where $\mathrm{t_i} = 1$ if individual i receives treatment. Thus $P(t_i|x_i)$ is the probability of receiving treatment conditional on covariates x_i. We can then use this weight to calculate the treatment effect using formula below:

$$\hat{\tau}_{ATE}^{IPW} = \frac{1}{n_{t_i=1}} \sum_{i:t_i=1} w_i y_i - \frac{1}{n_{t_i=0}} \sum_{i:t_i=0} w_i y_i$$

where $n_{t_i=1}, n_{t_i=0}$ are the number of observations in treatment group and control group respectively.

Doubly Robust Estimation: Intuitively, Doubly Robust Estimation improves Inverse Probability Treatment Weight by incorporating the linear relationships between the covariates and the outcome. As we discussed in section 3.2 and 3.3, even the propensity score is constructed correctly, we may still get the bias estimator. Therefore, we need to incorporate the linear regression into the estimator.

$$\hat{\tau}_{ATE}^{DRE} = \frac{1}{n} \sum_{i=1}^{n} \left\{ \left[\frac{y_i t_i}{\hat{P}(t_i|x_i)} - \frac{\tilde{y}_i^1 \left(t_i - \hat{P}(t_i|x_i)\right)}{\hat{P}(t_i|x_i)} \right] - \left[\frac{y_i(1-t_i)}{1-\hat{P}(t_i|x_i)} - \frac{\tilde{y}_i^0 \left(t_i - \hat{P}(t_i|x_i)\right)}{1-\hat{P}(t_i|x_i)} \right] \right\}$$

where $\tilde{y}_i^1, \tilde{y}_i^0$ are the estimated potential outcome of observation i.

In addition, we also want to examine how network size and covariates balance affect the performance of identifying treatment effect using representation learning. We separate the dataset into a two by two matrix: small (i.e. 10,000 users) vs large networks (i.e. 100,000 users) and balanced vs imbalanced data. We also want to note that because the models are estimated using a semi-synthetic data as described in section 4.2, this semi-synthetic data provides the

ground truth which can be used to compare the treatment effect estimated using different models. We list the results in Table 8 and 9. We can find that our deep representation learning (DRL) method outperform all three benchmark models in small scale network (10,000 users). In addition, even for unbalanced dataset, our DRL method can still outperform benchmark models. This is particularly useful for context in which there are many unobserved covariates that could affect the balance of the matched dataset.

The Table 8 shows except the MAE in sample, the Deep Representation Learning method can help to decrease the bias in the synthetic small and balanced covariates dataset on both the measurement of MAE and RMSE.

Table 8. Compare GNN methods with Propensity score-based method on treatment effect with small dataset and Balanced Covariates Measures (without GNN embeddings) on synthetic data

Small dataset with Balanced Covariates Features	MAE (In sample)	RMSE (In sample)	MAE (Out of sample)	RMSE (Out of sample)
Propensity Score Estimator	**0.024**	0.023	0.026	0.029
IPW Estimator	0.027	0.022	0.029	0.026
Double Robust Estimator	0.041	0.037	0.046	0.040
DRL	0.026	**0.021**	**0.024**	**0.026**

The Table 9 shows the Deep Representation Learning method can help to decrease the bias in the synthetic small and imbalanced covariates dataset on both the measurement of MAE and RMSE.

Table 9. Compare GNN methods with Propensity score-based method on treatment effect with small dataset and Imbalanced Covariates Measures (without GNN embeddings) on synthetic data

Small dataset with Imbalanced Covariates Features	MAE (In sample)	RMSE (In sample)	MAE (Out of sample)	RMSE (Out of sample)
Propensity Score Estimator	0.018	0.019	0.023	0.021
IPW Estimator	0.012	0.018	0.018	0.021
Double Robust Estimator	0.011	0.014	0.019	0.018
DRL	**0.009**	**0.007**	**0.014**	**0.016**

We list the results in Table 10 and 11. We can find that our deep representation learning (DRL) method outperform all three benchmark models in small scale network (100,000 users). In addition, even for unbalanced dataset, our DRL method can still outperform benchmark models. This is particularly useful for context in which there are many unobserved covariates that could affect the balance of the matched dataset.

The Table 10 shows the Deep Representation Learning method can help to decrease the bias in the big and balanced covariates dataset on both the measurement of MAE and RMSE.

Table 10. Compare GNN methods with Propensity score-based method on treatment effect with big dataset and Balanced Covariates Measures (without GNN embeddings) on synthetic data

Big dataset with Balanced Covariates Features	MAE (In sample)	RMSE (In sample)	MAE (Out of sample)	RMSE (Out of sample)
Propensity Score Estimator	0.035	0.032	0.041	0.037
IPW Estimator	0.028	0.032	0.031	0.035
Double Robust Estimator	0.029	0.026	0.033	0.029
DRL	**0.024**	**0.024**	**0.028**	**0.025**

The Table 11 shows the Deep Representation Learning method can help to decrease the bias in the big and imbalanced covariates dataset on both the measurement of MAE and RMSE.

Table 11. Compare GNN methods with Propensity score-based method on treatment effect with big dataset and Imbalanced Covariates Measures (without GNN embeddings) on synthetic data

Big dataset with Imbalanced Covariates Features	MAE (In sample)	RMSE (In sample)	MAE (Out of sample)	RMSE (Out of sample)
Propensity Score Estimator	0.048	0.045	0.053	0.046
IPW Estimator	0.041	0.037	0.044	0.039
Double Robust Estimator	0.038	0.034	0.039	0.037
DRL	**0.035**	**0.032**	**0.038**	**0.036**

4.4. Empirical Data Results

In this section, instead of using semi-synthetic data, we estimate the actual dataset using our proposed model, and demonstrate results in tables below. Please notice that we don't have the ground truth data now. What we can do is using the dataset with balanced covariates and 100,000 users. We compare our deep representation learning method with three other methods on this dataset.

Recall that we have Average Treatment Effect as:

$$\tau_{ATE} = E[Y(1) - Y(0)|X, A] = \frac{1}{n}\sum_{i=1}^{n}[Y_i(1) - Y_i(0)|X, A]$$

Table 12 and Table 13 shows that the deep graph representation learning method can always give the smallest treatment effect result compared with three other propensity score-based method. The deep graph representation learning is using graph neural network to construct the representation space in deep representation learning.

Table 12. Compare GNN methods with Propensity score based-method on treatment effect with small dataset and Balanced Covariates Measures (with graph features)

Small dataset with Balanced Covariates Features	ATE (In Sample)	ATE (Out of sample)
Propensity Score Estimator	0.322	0.328
IPW Estimator	0.382	0.397
Double Robust Estimator	0.290	0.302
DGRL(DRL with graph feature)	**0.254**	**0.289**

Table 13. Compare GNN methods with Propensity score based-method on treatment effect with large dataset and Balanced Covariates Measures (with graph features)

Large dataset with Balanced Covariates Features	ATE (In Sample)	ATE (Out of sample)
Propensity Score Estimator	0.435	0.452
IPW Estimator	0.414	0.436
Double Robust Estimator	0.315	0.362
DGRL(DRL with graph feature)	**0.328**	**0.347**

In table 14, we account the top 5 ranked hashtags as the treatment. They are ‘ANI’, 'Kirti Saxena', 'INC India', 'Kapil' , and 'With Congress'. This table shows that users are easily causal influenced by hashtag ‘INC India’ (0.524) but hardly causal influenced by hashtag ‘Kapil’ (0.197). This result is helpful to suggest the people who wants to increase the influence of tweets in social network.

Table 14. GNN based result with different treatment (hashtag)

Large dataset with Balanced Covariates Features	ATE (In Sample)	ATE (Out of sample)
Treatment #1	0.328	0.347
Treatment #2	0.342	0.376
Treatment #3	0.524	0.592
Treatment #4	0.197	0.211
Treatment #5	0.264	0.279

5. DISCUSSION AND CONCLUSION

Identifying the causal inference on the social network is an important and difficult question. This paper builds a graph-based representation learning method to identify the causal inference. We can find that the graph representation learning method partially solves the confoundedness problem that the propensity score method cannot solve very easily. The graph representation learning method extracts more graph embedding features that include the social network signals, which helps to identify the causal inference under social network circumstance.

We construct different types of datasets and compare our method with propensity score-based methods. The result shows that our model can always give robust results under small balanced, large balanced, small imbalanced, and large imbalanced covariates dataset. It also shows a robust result under different sample set.

The advantage of this graph representation learning method is it can solve the causal inference problem under large and imbalanced covariates social network. The disadvantage of this method is very consuming computing power. We don’t necessarily use it if we can construct the propensity score successfully.

ESSAY 2 GRAPH ATTENTION BASED METHOD FOR IMPROVING THE RECOMMENDATION FAIRNESS IN ELECTION SOCIAL NETWORK

0. ABSTRACT OF ESSAY 2

The graph based recommendation systems play an important role in social network platform. Most of these studies focus on building the appropriate representation space based on the interaction between users and items. However, this type of method is difficult to represent the information of the minority of users. This circumstance is called bias sampling in training process. The bias sampling on social network leads to the minority of users receive the unfairness recommendation. The graph structured based recommendation model even amplify this unfairness magnitudes. The existing graph embedding based recommendation algorithms focus on improving the recommendation quality but lose the attention on the fairness of these recommendation results. Very few studies pay attention on the fairness problem but difficult to give high recommendation accuracy with the fairness constraints. In this paper, we design an algorithm to satisfy both recommendation accuracy and fairness requirements. We present a graph attention based model to capture heterogeneous information between different users, tweets, and hashtags in twitter and use the attention mechanism to increase the disclosure of the sensitive attributes of the minority of users in twitter. The experiment shows that our model can make fair recommendation without losing too much recommendation accuracy. We compare our method with some benchmark methods and get the state-of-art performance.

1. INTRODUCTION

In 2014, The Bharatiya Janata Party (BJP) made a great success on general election and Nargendra Modi became the prime minister of the India. This is the first time when a party can win more than half seats in the Lok Sabha since last twenty years[6] (Priya and Peter 2014, Sardesai 2015, Ahmed et al 2016, Diwakar 2017, Khan et al 2019). Some political researchers pointed out that the reason behind BJP's success on election is to win the "Hindi Belt" (Diwakar 2017, Jaffrelot 2015, Khan et al 2019). Some other political researchers studied on the twitter account of the Nargendea Modi and declared it is not because BJP won the support but the Modi won the support (Sardesai 2015, Ahmed et al 2016, Pal et al 2019). If we analyze the content of the "naerndramodi": the official account of the Modi in twitter, there are many images such as zazen, meditation, and worship[7]. It helps Modi to build a religion leader image in his followers' mind (Khare 2015). Except building a religion leader image, Modi also posts texts, videos, and hashtags in his tweets. All these things can help Modi to interact with his followers very efficiently because his followers on twitter will know what is Modi's political declaration, political campaign, and persona (Ahmed and Skoric 2015, Khare 2015, Khan et al 2019). After these interactions between the Modi and his followers, the recommendation algorithm would recognize this kind of interactions as the preference of his followers and give the recommendation about the Modi's tweets content to the target followers in the future.

Further, politicians need to know the reactions and comments from their followers. The social network platforms such as twitter, facebook, and reddit are complement to traditional survey method (Berman et al 2019, Petrova et al 2021). It becomes more and more important for politicians to know how their followers' impressions to their tweets. The campaign team prefer

[6] https://www.indiavotes.com/lok-sabha/2014/all-states/16/0

[7] https://indianexpress.com/elections/pm-modi-takes-vacation-from-politics-meditates-in-kedarnath-5734996/

to recommend followers tweets based on their interests to attract more followers. (Pennacchiotti et al 2011) If we can extract the information from comments on twitter, it would be a good way to recommend users the political content about the politicians, parities, and shape candidates' image in followers' mind. Most of the recommendation algorithms in social network platform are based on these interactions between different users, hashtags, and tweets. The more interactions happened in the social network, the better patterns and preferences would be recognized by the recommendation algorithms.

This leads to the important question in recommendation systems on social network: not everyone's voice is heard equally in the social network (Dwork et al 2012, Feldman et al 2015, Hardt et al 2016). For example, there are many people believes in Hinduism in India. A muslim has very few chance to receive the information that Hinduisms always like to see in the twitter.[8] This is because the recommendation algorithms are based on the users' demographic data and the interactions between users and items. The majority of users have some preference of the tweets and it mislead the algorithm to build the stereotypes of the whole users without considering the minority of users (Rahman et al 2019). Therefore, the minority of users are represented by the majority of users and receive the unfair recommendation results. This is a sampling bias problem in the social network and cannot be avoided. The recommendation algorithms based these biased sample would always give the bias recommendation results, or unfairness recommendation results.

In order to solve these bias recommendation results, we need to build a model to consider both the recommended accuracy and the recommended fairness in the social network.

In this paper, we want to solve the following questions:

[8] https://www.pewforum.org/2021/06/29/religion-in-india-tolerance-and-segregation/

1. Which kind of information does politicians want to diffuse to the followers?

2. How to make sure the minority of users have chance to receive recommendation results based on their own preference in twitter?

3. How to build a recommendation algorithm satisfies both recommendation accuracy and the fairness in twitter?

In order to solve the above three questions, we combine three types of machine learning methods together. First, in order to know which kind of political information are always interesting to users' preference in twitter, we use the text mining methods, including bag of words and topic modeling (Karami et al 2018). It helps to describe the semantic information of the politicians' tweets, which includes the political declaration, political campaign, and persona. This is the information that politician want to impress on the followers' mind. Second, we need to detect how these semantic information diffuse between politicians and users' mind, the meta-path based method is an very efficient way to calculate it. In each meta-path, it describes how semantic information diffuse through different types of users, tweets, and hashtags. Therefore, we can build semantic meaning across different types of edges. When we consider the preference of the users of minority, this meta-path based method includes the many information from the users of minority and can somewhat decrease the sampling bias problem in the social network. Third, the reason why we have the bias sampling is some of the demographic features of the users are sensitive. It leads to the minority of users hard to receive the recommendation based on their own preference. In order to solve this problem, we put the attention mechanism on the minority of users' sensitive attributes and minimize the discrepancy between these attributes' representation space and the whole users' representation space. In this paper, we combine these three methods

together to capture the heterogeneous information on social network and balance off the recommendation accuracy and the recommendation fairness.

The traditional recommendation method is based on the collaborative filtering models. These models treat users and items (tweets) as pairs and give the recommendation based on the similar users' preference histories. However, it loses a lot of information about users' interaction. For example, two users on twitter may have no similar preference histories before but belong to the similar social network group (Bobadilla et al 2020). The interaction between users also reflect their interesting contents on twitter. Therefore, the traditional collaborative filtering models are not working well on the social network based recommendation. We need a recommendation model based on the graph theory.

Graph Representation Learning is a method to extract information from the large-scale network. (Kipf et al 2016, Hamilton et al 2017, He et al 2020) We can use the embedding method to encode both nodes and edges into the graph. The tweets posted by each user can be viewed as a node, and the retweet relationships between each two tweets can be viewed as an edge. We can use this method to draw a big semantic network based on the network of retweets. Please notice that this is not a social network or a text network. We call it semantic network because it includes tweets, users, and hashtags. These information are heterogeneous on each node and edge. We can use the meta-path based method to capture these heterogeneous information (Change et al 2015, Dong et al 2017, Fan et al 2019). The meta-path based model has many advantage on prediction accuracy in social network but has the limitation on considering the fairness (Bose and Hamilton 2019, Buyl and Bie 2021).

To solve the fairness problem, we need to build an unbiased representation space based on the biased observation (Fu et al 2020, Buyl and Bie 2021). We add the hierarchical attention

mechanism on the classical meta-path network embedding model. This method has many advantages: First, meta-path only give the global weights of each node and edge, but the attention can give us the local weights of each node and edge (Zhou et al 2019, Xue et al 2020, Wang et al 2021). Therefore, we can give the sensitive attributes of the minority of users more weights when we construct the representation space. Second, these attention weights can some how help to explain how different types of nodes and edges contributes to the final prediction accuracy and the fairness results (Dwork et al 2012, Wu et al 2019).

The contributions of this paper are as follows. First, we propose a model to embed the heterogeneous information network and amplify the representation of the minority of users in the twitter. Second, we use this method to the social recommendation problem and compare both accuracy performance and fairness performance with the benchmark methods and get the state-of-art results. Third, we get the evidence of the attention based on the attention coefficient, which can help to interpret the contributions of the focal nodes and edges.

The rest of the paper are written as follows: we review the literatures in section 2, declare the preliminaries and notations in section 3, build the model in section 4, analysis the experiment and data result in section 5, discuss and make the conclusion in section 6. The details of variable definition, model building, and additional results are in appendix.

2. LITERATURE REVIEW

There are three streams of studies to follow. The first stream is treating the political brand or political advertisement as a tool to help potential candidates to win the election (Gordon and Hartmann 2013, Wang et al 2018, Zhang and Chung 2020, Fossen et al 2021). Our study focuses on using the twitter as an election campaign tool to enlarge the political brand affection among the voters. The twitter is an important campaign tool in current world (Ahmed et al 2016, Diwakar 2017, Buccoliero et al 2020).

The second stream is the heterogeneous information network embedding based recommendation. This stream includes the heterogeneous information network embedding based recommendation and the graph neural network based recommendation. The heterogeneous information network embedding based recommendation is for capturing the different types of users, tweets, and hashtags and give the recommendation based on these information (Chang et al 2015, Fu et al 2017, Shi et al 2018). The attention mechanism from the graph neural network based recommendation is for capturing the weights for each specific intra-path or inter-path and give the recommendation based on each connection nodes or edges (Fu et al 2020, Hong et al 2020). We combine these two methods together to take the advantages both of them.

The third stream is the fairness recommendation. This stream includes the fairness sampling method and the fairness modeling method (Dwork et al 2012, Zeng et al 2021). The fairness sampling method means making the sampling not depends on the sensitive attributes. The fairness modeling method means making the fairness loss function during the training process in order to take care the preference of the users of minority (Bose and Hamilton 2019, Bobadilla et al 2020, Buyl and Bie 2020, Spinelli et al 2021). Both of these two ways can help to decrease the fairness of the recommendation.

We fill the gap between heterogeneous information network embedding and fairness recommendation with attention mechanism. To the best of our knowledge, this is first study of using the attention mechanism to decrease the bias under recommendation systems.

Therefore, we contribute on both the theoretic part and the methodology part. In the theoretic part, we prove the twitter is an efficient tool for the politicians and their parties to make the campaign. It helps to improve the politician impression on users' mind. In the methodology part, we contribute an advanced model to calculate the representation considering the minority of users' preferences and their interactions in social network platform. This method can give fairness recommendation to the minority of users without losing too much recommendation accuracy.

2.1. Political Brand Image on semantic network

The twitter is a growing fast platform to share different opinions and public views related to political things. It can enlarge the positive and negative images of the political candidates in followers' mind. Twitter is an very important tool to win the campaign and predict the voters' opinions (Bermingham and Smeaton 2011, Skoric et al. 2012, Sang and Bos 2012, Berman et al 2019, Khan et al 2019, Petrova et al 2021). Many users studied the social media platform such as Twitter and Facebook to extract the semantic meaning behind the text information and find the connection between these semantic meaning and the political campaign success (Ahmed and Skoric 2015).

The India's 2014 general election result changes the voting behavior of the India. (Diwakar 2017, Khan et al 2019). Many people becomes to follow the trend and interact with potential candidates in twitter. The BJP and its leader Modi, successfully transformed the followers to voters. They build a concept called "Modi lahar" to attract many followers (Diwakar 2017, Khan

et al 2019). Modi and his campaign team successfully built a political image as a fighter and a lord to lead Indian people.

Ahmed et al (2017) used political brand image as the mediator and found there is a positive relationship between the political brand and the voting preference. Lin and Himelboim (2019) also used political brand as the mediator and found that the weaker social ties of the political brand community, the more voters will be attract to the political candidates. The political brand is also used as the moderator in the political brand research. It strengths the relationship between the tweet content and the voters' engagement in twitter (Panda et al 2020, Mallipeddi et al 2021) .

Except the brand image, some other researchers studied the political advertisement. (Graham et al 2013, Gordon and Hartmann 2013, Zhang and Chung 2020, Fossen et al 2021) They found that the political advertisement has significant positive relationship between the election results. The more effectiveness of the candidates' advertisement, the more voters would support the candidates. Therefore, we can see the political advertisement is very useful to get more votes. In current time, twitter becomes more and more important to interact with the voters. More and more politicians treat twitter as a platform to advertise themselves.

No matter building the political brand image or increasing the political brand advertisement effect, the campaign team of the political candidates should use twitter carefully. The twitter is an important channel to diffuse political candidates' opinions (Berman et al 2019, Panda et al 2020, Petrova et al 2021). If the recommendation algorithm recognize these interactions between the voters and political candidates well, the information will diffuses very efficiently in the twitter.

Table 15. Literature review of Political Brand Image on semantic network

Author	Political Marketing	Conclusion
Ahmed et al (2017)	Political Brand Image as the Mediator and Predictor	Political Brand Equity has positive relationship with the voting preference
Lin and Himelboim (2019)	Political Brand Community as the Mediator	The weaker social ties of the Political Brand Community, the more successful of the Political Candidates
Khan et al (2019)		
Mallipeddi et al (2021)	Political Brand as the Moderator	Political Brand Image moderates the correlation between the content and the voters' engagement
Gordon and Hartmann (2013).	Political Advertisement as the Predictor	The Political Advertisement is positive correlation with the presidential elections
Wang et al (2018)	Political Advertisement as the Mediator	The effectiveness of the political advertisement sponsored by Political action committees is less than sponsored by the political candidates
Zhang and Chung (2020)	Political Advertisement as the Predictor	The effectiveness of the political advertisement from the political candidates is more than the political advertisement from the non-partisans
Fossen et al (2021)	Political Advertisement as the Predictor	Political Advertisement has the spillover effect on the subsequent advertisement
Berman et al (2019)	Twitter as the online advertisement channel	Tweets becomes more emotionally and elaborate the debates
Petrova et al (2021)	Facebook as the online advertisement channel and donation channel	The higher twitter penetration area, the higher donation for the political candidates

2.2. Heterogeneous Information Network Embedding

In recent years, graph neural network (GNN) is a popular representation method to extract node, edge, and graph features from the social network (Xie et al 2016, Hamilton et al 2017, Zhang et 2019). It can help us to analyze the users' behaviors and the semantic information effectively. There are two types of embedding methods to learn the graph features. One is the random walk based method, another one is the graph neural network based method. The advantage of using the random walk based model, the meta-path embedding, is to capture all the heterogeneous information easily (Dong et al 2017, Fan et al 2019, Yin et al 2019). The advantage of using the graph neural network, especially the attention mechanism, is to aggregate the intra-path and inter-path information easily (Sankar et al 2018, Xue et al 2020). Therefore, in this paper, we combine the random walk based method and the graph neural network based method together to train the model.

This section includes three parts. First, the heterogeneous information network embedding. There are many researchers studied the heterogeneous network, especially use the meta-path based method to embed the features on the same path (Ji et al 2018, Shi et al 2018). Second, the graph neural network. It includes graph convolution network aggregation way and graph attention aggregation way (Nguyen et al 2018, Pareja et al 2020, Sankar et al 2020). Both of them have advantage to train the model but the attention mechanism is easy to connect with the meta-path based method. Third, the graph neural network based recommendation. After we combine these two methods, we need to apply it on the recommendation problem. It has better performance compared with the traditional recommendation method (Wu et al 2018, Fan et al 2019).

We review the meta-path based papers and attention based papers. Then we discuss how to combine them together and apply it into the social recommendation problem.

There are many users, topics, tweets and hashtags in twitter. If we treat these different types of nodes as the homogeneous graph, we will lose the information when we transform them into the embedding space. This will lead to failure if we use the embedding results to do the classification, prediction, and recommendation problems. Therefore, we need to use a representation method to capture all these heterogeneous nodes together.

Meta-path method, which is based on the skip-gram framework, can successfully includes different types of nodes and their semantic meanings. We can construct the information diffusion path based on the semantic meanings and calculate the highest probability from these path instances. Dong et al (2017) firstly proposed the meta-path method to build the diffusion path and capture the features from different types of nodes. Fu et al (2017) relaxed the restriction and construct the meta-path more flexible. Shi et al (2018) followed the same idea and proposed HERec method. They defined a filter to capture the node sequence and transform the heterogeneous topology network to the homogeneous topology network.

Except using the meta-path based methods, Many researchers extracted the different types of features with graph neural network method (Kipf and Welling 2016, Zhang et al 2019, He et al 2020, Fu et al 2020, Wang et al 2021). Kipf and Welling (2016) and He et al (2020) used graph convolutional network methods to construct the embedding space. However, when we use the graph convolutional network, we aggregate the information from the neighbor nodes of the target nodes. Therefore, we cannot embed the predefined diffusion path information into the model. In this paper, we plan to give more explanation of the model results. The graph convolutional network method cannot give us the apparently explanation of the results. We choose another way, the graph attention network (Zhou et al 2019, Fu et al 2020, Wang et al 2021) to aggregate the information. Zhou et al (2019) combined the attention mechanism and the meta-path method

together. In node level, each node features are aggregated by the attention mechanism. It still contains the meta-path structure and aggregate the information by neural network. Fu et al (2020) extend this method by aggregate the information on different edge level. Therefore, we can capture the features from both different types of nodes and different types of edges simultaneously. This type of method needs to predefine the meta-path before we train the model. Wang et al (2021) only use the attention mechanism to extract the different types of nodes, edges, and side information but not use the meta-path.

Except the pure attention mechanism, some other researchers modify the attention mechanism with other embedding method (Wu et al 2019, Fan et al 2019, Zhang et al 2019, Hong et al 2020, Xue et al 2020). For example, Wu et al (2019) studied the social recommendation problem based on the dual graph structure. All of the attentions are calculated from the different nodes and aggregate on different graphs. Zhang et al (2019) replaced the attention mechanism by the LSTM embedding method. After discussing these papers, we can see that using the attention mechanism to aggregate different types of information is an effectively way.

We propose a model based on the meta-path method and attention mechanism to capture the heterogeneous information on twitter. By using the attention mechanism to aggregate the graph information on each meta-path, we can represent the information of different users, hashtags, tweets, and topics on the twitter. We reviews the papers how can we use the attention mechanism to decrease the bias of recommendation in the next part.

Table 16. Literature review of Heterogeneous Information Network Embedding

Method	Author	Model Name	Node level Aggregation	Edge level Aggregation
Meta-path based Methods	Dong et al(2017)	Metapath2vec	Meta path	NA
	Fu et al (2017)	HIN2Vec	Meta path	NA
	Shi et al (2018)	HERec	Meta path	NA
Hierarchical Attention Mechanism	Wang et al (2021)	HAN	Node Attention	Edge Attention
	Zhou et al (2019)	HAHE	Meta path Attention	NA
	Fu et al (2020)	MAGNN	Meta Path Attention	Meta Path Attention
	Wu et al (2019)	DANSER	Dual Attention	NA
	Fan et al (2019)	MEIRec	Meta Path Attention	NA
	Zhang et al (2019)	HetGNN	LSTM	Edge Attention
	Hong et al (2020)	HetSANN	Hierarchical Attention	NA
Our Model		Attention Fair	Intra-Path Attention	Inter-Path Attention

2.3. Fairness recommendation algorithms

Despite the success of using graph neural network to train the data, it also leads to the bias on sensitive attributes such as age, gender, and religion. The representation of the graph neutral network amplify the bias of the especially for the minority of users. (Rahman et al 2019, Bose and Hamilton 2019, Bobadilla et al 2020, Fu et al 2020, Wang et al 2021)

There are two ways to decrease the bias during training. First, we can build a sampling method considering about the minority people when we generate the embedding space. Most of papers in this stream follows Rahman (2019) and Buyl and Bie (2020) ideas to calculate the specific fairness based random walk. This type of method can help to improve the fairness of the recommendation but loss the accuracy of the prediction. Zeng et al (2021) modified the traditional meta-path generation method by giving high weight for minority group and low weight for majority group. This can help to reduce the sampling bias during embedding process. Wu et al (2021) detected the sensitive attributes at first step, and built a filter space to decrease the bias of each sensitive attributes. Therefore, the filtered embedding space can give the fairness recommendation result compared with no-filtered embedding space.

Second, we can build the model considering about the fairness and put it into the loss function. (Fu et al 2020, Wang et al 2021) This can help to trade-off the accuracy of the prediction and the fairness of the recommendation. The sampling bias can be eliminated based on the generated adversarial network. Dai and Wang (2021) used an adversarial neural network to get unbiased the node's sensitive features. This method can help to decrease the fairness when we construct the graph neural network. However, it also lose some generalization of the model (Kang and Tong 2021). In order to solve this problem, many studies construct the loss function based on the ranking method (Fu et al 2020, Spinelli et al 2021, Dong et al 2021). Fu et al (2020) put the relative ranking of the users of minority into the Gini Index formula to measure the unfairness. Spinelli et al (2021) built a graph neutral network to drop the biased edge after each training epoch. The biased edges are based on some relative distance metrics. Dong et al (2021) promoted the fairness of individual by considering the distance matrix of each user and item pairs. They built a ranking based loss function. If the relative ranking distance of the minority of

the users is different with the majority of the users in the ranking space, they will change the ranking and put it into the next turn training.

Our model choose the second way to solve the fairness problem. We use the attention mechanism to enhance the preference of the minority of users' attributes but not the majority users' attributes. The idea of this minimize the differences between the minority of users' preference and the majority of users' preference can be found in some studies using the KL divergence (Buyl and Bie 2021, Current et al 2022). Our method share the same idea and we put this idea into the graph neural network architecture. It can help to consider the fairness of the recommendation and not lose much accuracy of the prediction.

Table 17. Literature review of Fairness recommendation algorithms

Author	Model Name	Sampling or Modeling	User level or Group level	Neutral Network Architecture
Rahman et al (2019)	Fairwalk	Sampling	User level	NA
Fu et al (2020)	FairKG4Rec	Modeling	Both	NA
Dai and Wang (2021)	Fair GNN	Modeling	User level	GAN
Spinelli et al (2021)	Fair Drop	Modeling	User level	GNN
Zeng et al (2021)	Fair HIN	Sampling	Group level	GAN
Wu et al (2021)	Fair Go	Sampling	Group level	GAN
Dong et al (2021)	EDITS	Modeling	Group level	GNN
Dong et al (2021)	REDRESS	Modeling	User level	GNN
Our model	**Attention Fair**	**Modeling**	**Both**	**GNN**

3. PRELIMINARIES

After we review the prior studies of the heterogeneous information network embedding and the fairness of the recommendation. We give the preliminaries of the concept in this section.

Definition 1 Heterogeneous Information Network

The heterogeneous networks is defined as G, where $G = (V, E)$. The V denotes the original set of nodes and E denotes the original set of edges. There are two mapping functions, node type mapping function: $f_v: V \to R$ and edge type mapping function $f_e: E \to S$. The R are the predefined set of node types and S are the predefined set of edge types. Where $|R| + |S| > 2$

The definition of the heterogeneous information network is given by the Dong et al (2017)

Definition 2 Heterogeneous Information Network Embedding

Given a heterogeneous graph G, where $G = (V, E)$, with node types R and node attributes matrix $X_T \in \boldsymbol{\mathcal{R}}^{V_T \times d_T}$, where T is the number of types. The Heterogeneous Information Network Embedding is a task to learn d-dimension node representations $h_T \in \boldsymbol{\mathcal{R}}^{\boldsymbol{d}}$ for all $v \in V$ such that capture graph information as more as it can. If $T=1$, it becomes a homogeneous information network.

The definition of the heterogeneous information network embedding is given by the Dong et al (2017)

Definition 3 Meta-path

A meta-path ϕ is defined as $\phi = \mathrm{R}_1 \xrightarrow{S_1} \mathrm{R}_2 \xrightarrow{S_2}, \dots, \xrightarrow{S_m} \mathrm{R}_{m+1}$, A path instance I is a path go through the node $V_1, V_2, \dots, V_m$ where, $i = 1, \dots, m$, $R_i = f_v(V_i)$ and $S_i = f_e(V_i, V_{i+1})$

The definition of specific meta-path needs the prior knowledge. Although there are studies using unsupervised way to calculate the meta-path in social network (Wang et al 2018, Wei et al 2018). We need to specific the sensitive attributes in the fairness problem. Therefore, we predefine the

meta-path based on the semantic information from each tweet, user and hashtag. The details of predefinition can be found in Appendix K.

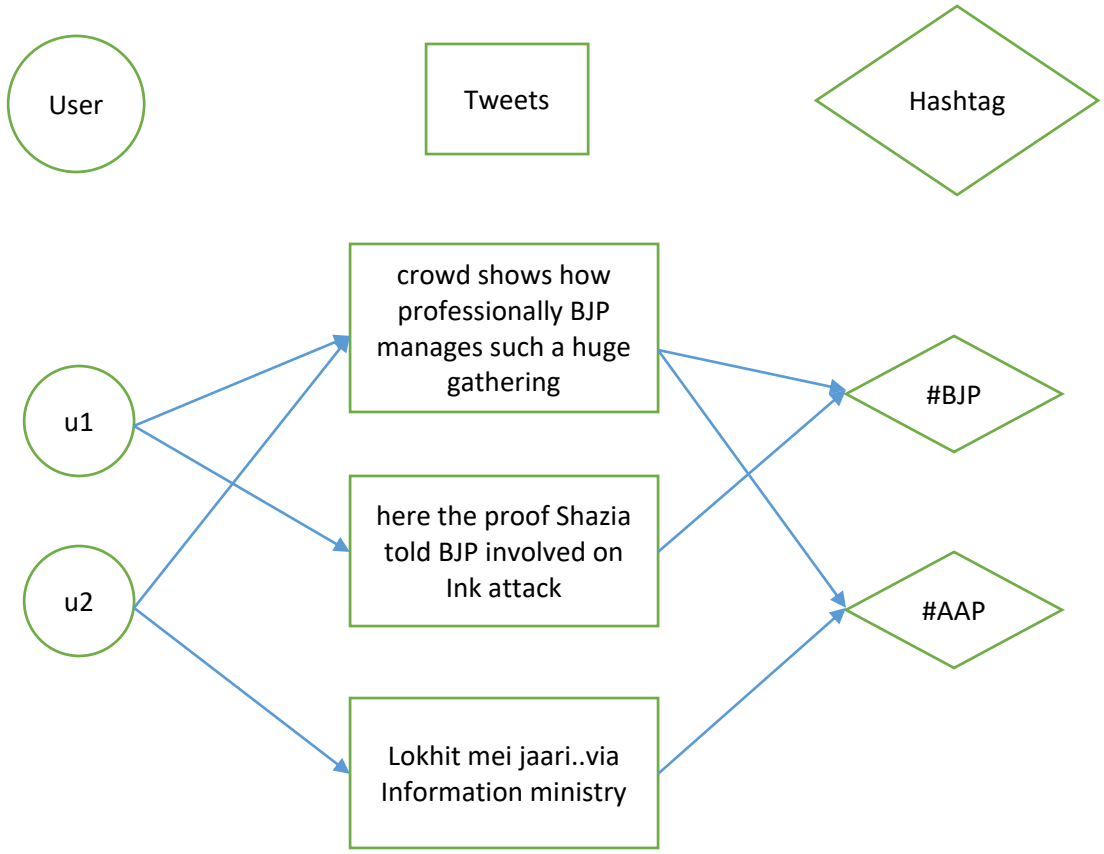
Definition 4 Meta-path based Neighbors

Given a node V_i and a meta-path ϕ in a heterogeneous graph, the meta-path based neighbors N_ϕ is the neighbor nodes connect to node i by meta-path

After making the definition of the meta-path, we draw some diagrams to illustrate as examples of the meta-path:

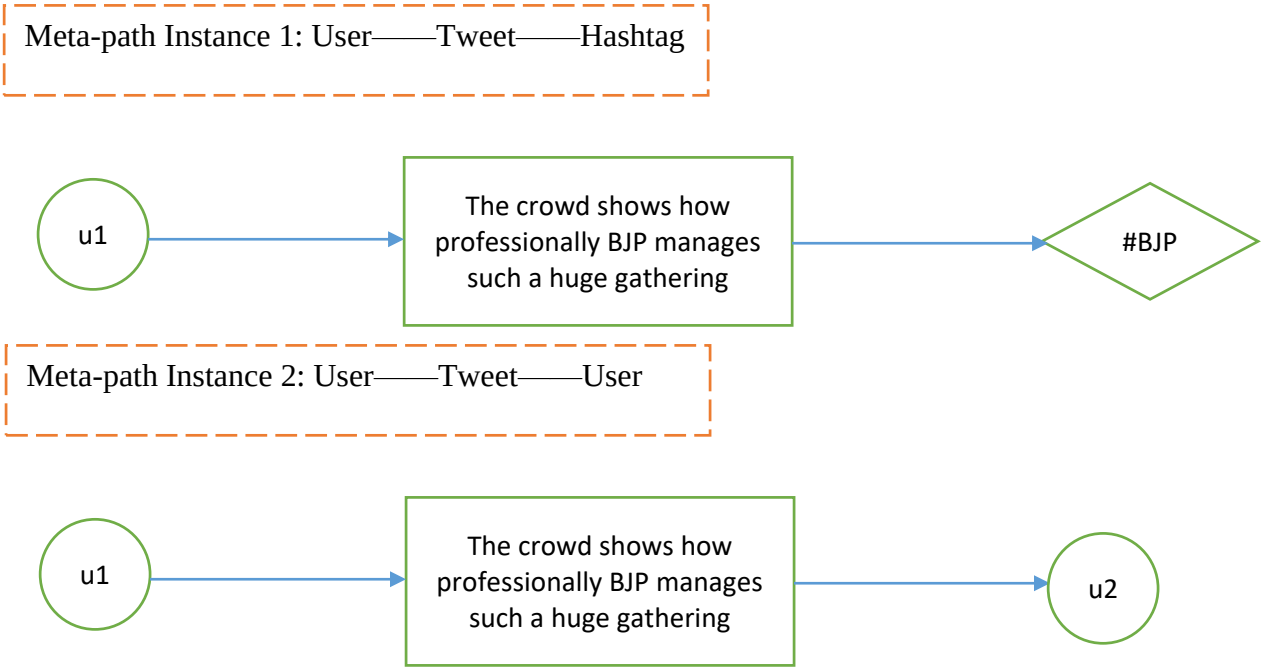
In figure 3.1, it shows the diagram of how opinion diffusion in twitter. The user 1 retweets the content as “crowd shows how professionally BJP manages such a huge gathering” with the hashtag BJP included. This diffusion connect different types of users: u1 and u2, the same tweet, and two different types of hashtags: #BJP and #AAP. This could be seen as a local heterogeneous information network with different types of users, tweets, and hashtags. For each different types of users, we can define different types of edges to connect these nodes.

Figure 6. Diagram of opinion diffusion in twitter



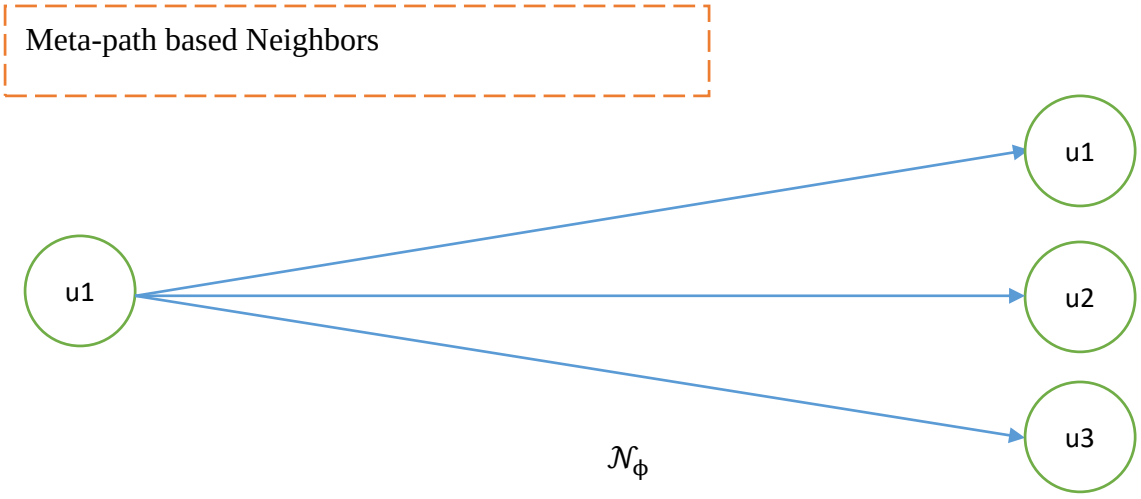
In figure 7, we build the meta-path instance from the heterogeneous information network in figure 6. We have two types of meta-paths to illustrate. The first meta-path is the diffusion from u1 to tweet1 to hashtag1, written as User-Tweet-Hashtag and denoted as Φ_1. The second meta-path is the diffusion from u1 to tweet1 to u2, written as User-Tweet-User and denoted as Φ_2. Obviously, this is a simple example to illustrate how we define the meta-path. We can define more types of users, tweets, and hashtags to reveal more information. The details of these definition can be found in Appendix L.

Figure 7. Meta-path Instances



In figure 8, we build the meta-path based neighbors for the focal node u1. In the meantime, we can find the meta-path based neighbors such as u1, u2, and u3 based on the meta-path that connect all these three users together and we denote it as $\mathcal{N}_{\phi}$. We can also define the meta-path based neighbors for tweets and hashtags. In this paper, we focus on solving the fairness problem of the minority of users in twitter. Our meta-path based neighbors are only for users' neighbors.

Figure 8. Meta-path based Neighbors



Except introducing the preliminary knowledge of the heterogeneous information network, we also need to define the measurement of the fairness. In economics and sociological studies, Gini coefficient is always used to measure the fairness of people's income (Gini 1921). In this setting, we use Gini coefficient to measure the fairness of the recommendation quality to users in social network. This is very helpful to measure the individual fairness for each user.

Definition 5 Gini Coefficient

Suppose we have m users $u_1, u_2, \dots u_m$ and n recommended tweets $tw_1, tw_2, \dots tw_n$. We denote $Q_{ij} = \{0,1\}$ as whether tweet tw_j is recommended to user u_i The top-K recommendation means user i can receive the top K recommended tweets tw_j, which is denoted as $\sum_{j=1}^{n} Q_{ij} = K$ The Gini coefficient measurement in this social network setting is written as:

$$Gini(Q) = \frac{\sum_{Q_\mu, Q_\nu} |Q_\mu - Q_\nu|}{2m \sum_{j=1}^{n} Q_\mu}$$

Where μ and ν are two random users selected from the whole users.

There are another two ways to measure the group fairness to users with similar sensitive attributes such as age, gender, and religion in social network.

Definition 6 Demographic Parity

Demographic Parity is one of the important criteria of fairness given by Dwork et al (2012). It means the probability of being assigned to the predicted result $\hat{y}$ (For example, the recommended engineering book in Amazon) should be independent to the sensitive attributes $x_{sensitive}$. Suppose we select one user u_μ belong to the user of minority. (For example, the sensitive attribute is female). We select another user u_v belong to the user of majority (For example, the sensitive attribute is male). We expect the user should get engineering book recommendation in Amazon not because of gender. The formula is written as:

$$P(\hat{y} = 1|x_{sensitive} = 0) = P(\hat{y} = 1|x_{sensitive} = 1)$$

Definition 7 Equal Opportunity

Another important criteria of the fairness is defined by Hardt et al (2016), called Equal Opportunity. It means the probability of the group of people is assigned to the predicted result $\hat{y}$ should be independent to the sensitive attributes $x_{sensitive}$ and the corresponding group truth result y. We expect the user should get engineering book recommendation in Amazon not because of gender and in the meantime, they really like to get engineering book recommendation. The formula is written as:

$$P(\hat{y} = 1|x_{sensitive} = 0, y = 1) = P(\hat{y} = 1|x_{sensitive} = 1, y = 1)$$

The advantage of using the equal opportunity is, we consider a situation that a sensitive group of users may get recommendation because of other reasons (Hardt 2016). For example, if a female has many friends belong to the engineering major. The algorithm will predict this female has strong interesting on the engineering. However it is not true. Therefore, we need to put the

ground true outcome as the condition to predict the recommendation result. Compared with Demographic Parity, this measurement put more weights on the true positive rate. It can help to avoid the unfairness situation that the users belong to the minority group truly but assigned as majority falsely.

After we give all these preliminary statements, we discuss the model building and data analysis in next two sections.

4. MODEL SETTING AND ANALYSIS

4.1. Model setting

We can set the model as follows, a user i belongs to a graph $G(V, E)$, where V is the vertex, E is the edge of this graph and A is the corresponding adjacency matrix. Each user i has demographic information, notated as X, called node attributes. Each user i posts some tweets or retweets on the twitter including the sentiment scores y, which is notated as Y, the label data for each node. Some papers put the attention mechanism on nodes and edges to aggregate the information (Wu et al 2019, Xue et al 2020, Wang et al 2021). It is not a good way to solve the fairness problem because it give more weights on the majority of users' demographic attributes and make the recommendation result even more unfairness. Therefore, we use the attention mechanism on the meta-path to aggregate the information. This helps to solve the problem that the minority of users are represented by the majority of users in the social network. After each aggregation in each meta-path, we make the aggregation on different meta-path, this helps to construct better representation about the heterogeneous information in social network (Fan et al 2019, Sankar et al 2019, Yin et al 2019). In the end, we still need to build a loss function to control the fairness of the minority of users. However, we don't follow the method as previous because they put more weight on the fairness bias or the equal opportunity (Dai and Wang 2021, Wu et al 2021, Zeng et al 2021). We still want to preserve the recommendation accuracy. Therefore, we build the same attention mechanism architecture on the minority of users as what we train in the whole dataset. We build a loss function to minimize the distance between the minority of users' representation and the whole users' representation. If their distance is as low as possible, it mean the minority of users are not represented by the majority of users in the training process. Therefore, the training algorithm can satisfy both high recommendation results and the fairness of the minority.

The process of building our model is as follows: At first, we transform the node in to the representation space. Second, we aggregate information of each different types of nodes include user, tweet, and hashtag belong to the same semantic meta-path. Third, we aggregate information of each meta-path and output the whole semantic representation space. Fourth, we calculate the cross-entropy of the final embedding space with the corresponding labels. In the mean time, we use the same training process on the minority of users and calculate their final embedding space. If we can minimize the distance between the final embedding space of whole users and the final embedding space of minority of users, we can get the relative fairness training result from the model.

4.2. Intra-Path Attention

The node-level attention is to calculate the important weight of each node's neighbors and itself. The project from node into feature vector

$$h_i = W^r \cdot x_i^r$$

Where x_i^T is the initial feature for node i with type r, W^r is the transformation matrix for node type r.

The meta-path embedding is written as

$$h_{\phi(i,j)} = f_\theta(h_i, h_j, \forall i, j \in \{N_\phi\})$$

If we calculate the attention between two nodes h_i and h_j, we can get the following attention

$$e_{ij} = att_{node}(h_i, h_j)$$

We modify it with including the meta-path ϕ as

$$e_{ij}^{\phi} = LeakyReLU(a^T \cdot [h_i || h_{\phi(i,j)}])$$

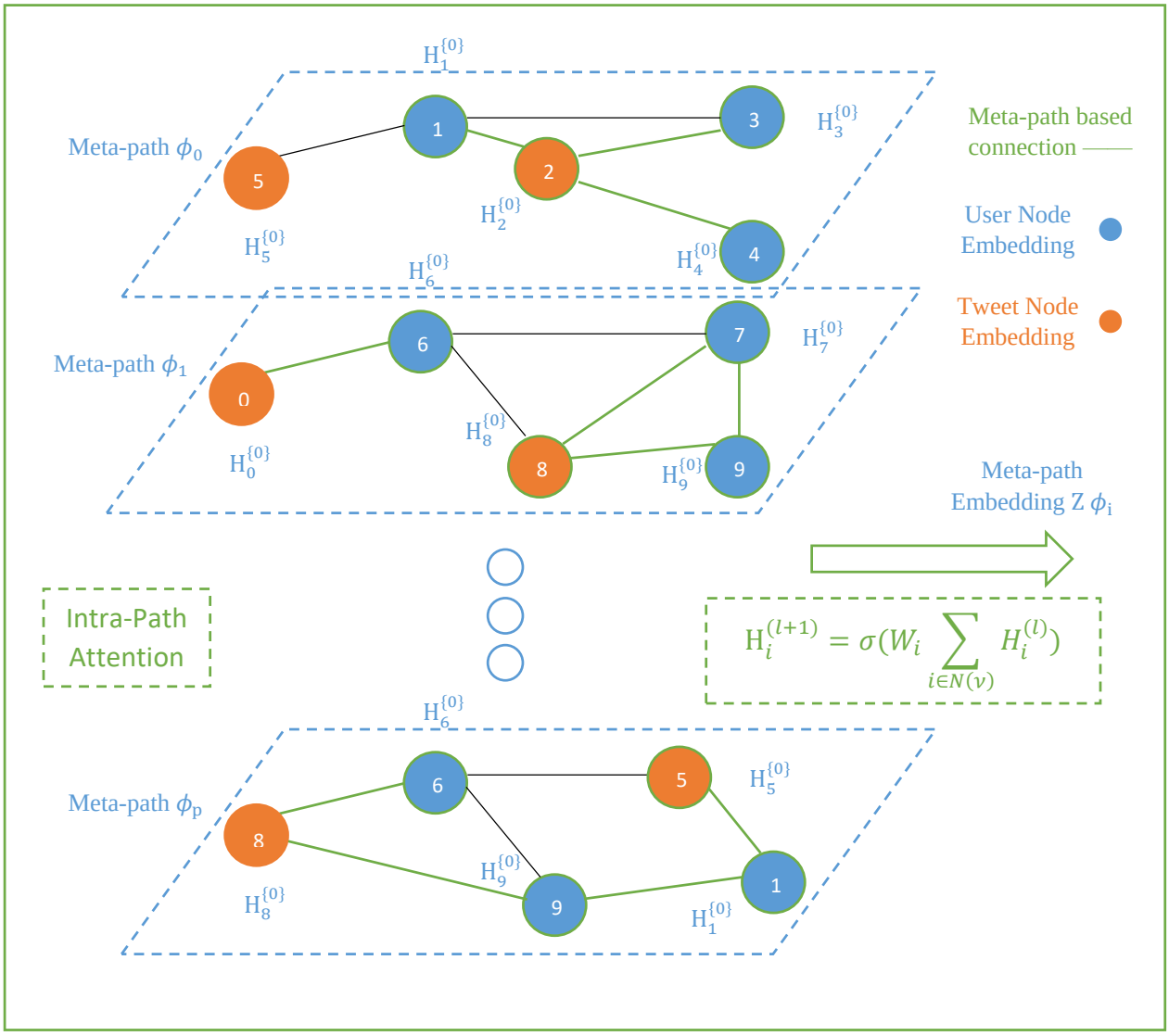
The weight of two nodes i and j for edge type r is calculated as

$$\alpha_{i,j}^{r} = softmax_j(e_{ij}) = \frac{exp(\sigma(a_r^T \cdot [W^r \cdot x_i || W^r \cdot x_j]))}{\sum_{k \in N_i^{rt}} (exp(\sigma(a_r^T \cdot [W^r \cdot x_i || W^r \cdot x_k]))}$$

N_i^r is the neighbors of node i with edge type r.

a_r is the parameter weight of the attention, σ is the activation function, and || is the concatenation operation.

Figure 9. Diagram of the Intra-Path Attention



We can get the final representation as below

$$h_i^r = \sigma(\sum_{k \in N_i^r} \alpha_{i,j}^r \cdot W^r x_j)$$

Extend to multi-head attention

$$h_i^r = \mathop{\Big\|}_{k=1}^{K} \sigma(\sum_{k \in N_i^r} \alpha_{i,j}^r \cdot W^r x_j)$$

4.3. Inter-Path Attention

After we calculate the node-level information, we need to aggregate the information of each edge. We call it the semantic-level information. The semantic-level attention of node I for edge type r is calculated as

The important of each semantic is calculated as

$$w = \frac{1}{|V|} \sum_{i \in V} q^T \cdot tanh(W \cdot h_i^r + b)$$

$$\beta_t^r = softmax(w) = \frac{exp(q^T \cdot \sigma([W \cdot h_i^r + b]))}{\sum_{r \in R}(exp(q^T \cdot \sigma([W \cdot h_i^r + b]))}$$

We can get the final representation as below

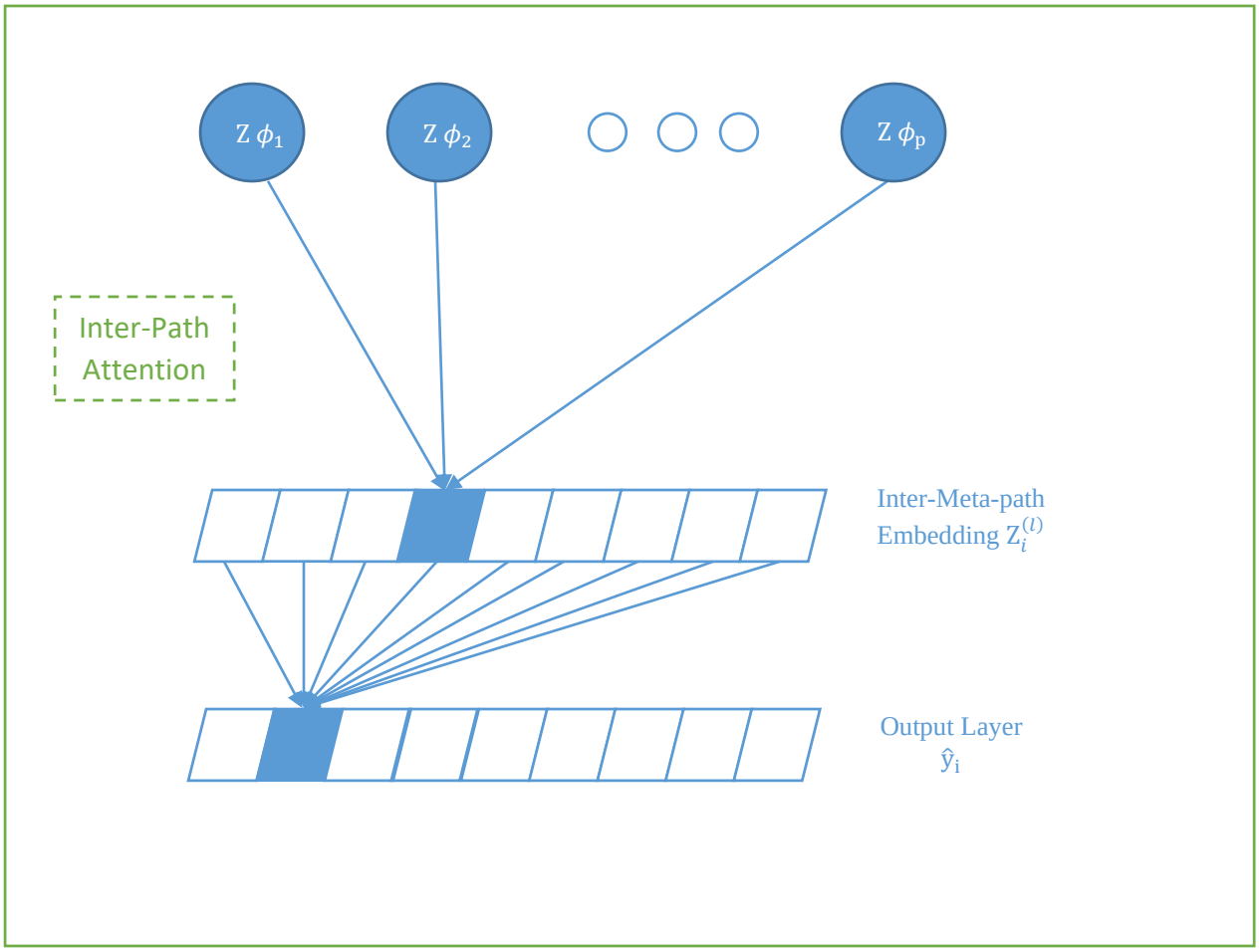
$$Z_i = h_i \ = \sum_{r=1}^{R} \beta_i^r \cdot h_i^r$$

The object function can be written as minimize the cross-entropy over all node

$$L_{accuracy} = -\sum_{c=1}^{C} \sum_{i \in V} y_i[c] \cdot log\, Z_i[c]$$

Where the V is the set of nodes that have labels , C is the number of class and y_i is the ground truth.

Figure 10. Diagram of the Inter-Path Attention



4.4. Attention Fairness

Intra-Path Attention for the minority of users

The node-level attention is to calculate the important weight of each node's neighbors and itself.

The project from node into feature vector

$$h'_i = W'^r \cdot x'^r_i$$

Where x_i^T is the initial feature for node i with type r, W^r is the transformation matrix for node type r. The meta-path embedding is written as

$$h'_{\phi(i,j)} = f_\theta(h'_i, h'_j, \forall i,j \in \{N_\phi\})$$

If we calculate the attention between two nodes h'_i and h'_j, we can get the following attention

$$e'_{ij} = att_{node}(h'_i, h'_j)$$

We modify it with including the meta-path ϕ as

$$e'^{\phi}_{ij} = LeakyReLU(a'^{T} \cdot [h_i || h_{\phi(i,j)}])$$

The weight of two nodes i and j for edge type r is calculated as

$$\alpha'^{r}_{i,j} = softmax_j(e'_{ij}) = \frac{exp(\sigma(a'^{T}_{r} \cdot [W'^{r} \cdot x'_i || W'^{r} \cdot x'_j]))}{\sum_{k \in N'^{rt}_i}(exp(\sigma(a'^{T}_{r} \cdot [W'^{r} \cdot x'_i || W'^{r} \cdot x'_k]))}$$

N'^{r}_{i} is the neighbors of node i with edge type r.

a'_r is the parameter weight of the attention, σ is the activation function, and || is the concatenation operation.

We can get the final representation as below

$$h'^{r}_i = \sigma(\sum_{k \in N'^{rt}_i} \alpha'^{r}_{i,j} \cdot W'^{r} x'_j)$$

Extend to multi-head attention

$$h'^{r}_i = \mathop{\Big\|}_{k=1}^{K} \sigma(\sum_{k \in N'^{rt}_i} \alpha'^{r}_{i,j} \cdot W'^{r} x'_j)$$

Inter-Path Attention

After we calculate the node-level information, we need to aggregate the information of each edge. We call it the semantic-level information. The semantic-level attention of node i for edge type r is calculated as

The important of each semantic is calculated as

$$w' = \frac{1}{|V|} \sum_{i \in V} q^T \cdot tanh(W' \cdot h'^{r}_i + b)$$

$$\beta'^{r}_{t} = softmax(w') = \frac{exp(q^T \cdot \sigma([W \cdot h'^{r}_{i} + b]))}{\sum_{r \in R}(exp(q^T \cdot \sigma([W \cdot h'^{r}_{i} + b]))}$$

We can get the final representation as below

$$Z' = h'_i = \sum_{r=1}^{R} \beta'^{r}_{i} \cdot h'^{r}_{i}$$

The fairness loss function is defined as the distance between the representation of the all users and the representation of the minority of users.

$$L_{fairness} = dis(Z, Z')$$

4.5. Training

In order to train the model considering both accuracy and the fairness, we combine two loss function together and get the final loss function as

$$L = L_{accuracy} + \lambda * L_{fairness}$$

The objective is to minimize the cross-entropy between the group-truth label y and the predicted representation Z and minimize the distance between the predicted representation Z from whole users with the predicted representation Z' from the minority of users. We optimize the model with mini-bath stochastic gradient descent and back propagation. The overall description of the model is in Algorithm 1.

4.6. Model Analysis

There are many advantages for our model. Fist, the proposed model can handle many types of nodes, edges, and features.

Second, this model is easily compute and parallelized. The overall complexity is linear to each meta-path pairs and can be parallelized across different nodes types and meta-path pairs. The time complexity for each intra-path loop is $O(V_\phi KL^2)$ and for each inter-loop is $O(E_\phi KL)$.

Third, this model is good for explain the contribution from the focal nodes and edges. When we calculate the attention weight coefficient, it will show us how does it contribute to the final task. Therefore, it is easy to investigate which part is important on the whole graph.

Algorithm 1. Attention Fair Algorithm

Input: $G^t = \{V^t, E^t\}$, meta-path set $\{\phi_1, \phi_2, \dots, \phi_p\}^t$, features set $\{x_i, \forall i \in V\}^t$
Output: Final Embedding Z, intra-path attention coefficient α, inter-path attention coefficient β

For $\phi_i \in \{\phi_1, \phi_2, \dots, \phi_p\}$ **do**
  For k=1,2,…K **do**
    Intra-path embedding
    For $i \in V$ **do**
      Find the meta-path neighbors $\mathcal{N}_\phi$
      For j $\in \mathcal{N}_\phi$ **do**
        Calculate the intra-path weight coefficient
      End
    End
    Calculate the semantic embeddings
  End
  Calculate the inter-path weight coefficient
  End
  Calculate the semantic embeddings
End
Calculate the final embeddings
Calculate the cross-entropy
Back propagation and update the parameters
Return Z, α, β

5. DATA RESULT

5.1. Dataset: Indian 2014 Election Twitter

Our dataset is the Indian General Election data from the Twitter between Jan 1st and May 16th 2014. This dataset contains all tweets related to Indian General Election. This dataset contains 3.2 millions tweets, from 218, 7334 users. After data cleaning, we can get the tweets sentiment, *Y*, the treatment *T*, (i.e. whether user i is exposed to others' tweets of the same topic before she posts m^{th} tweet), the covariates variables *X* such as User Location, User Followers Count, etc, and the adjacency matrix *A*.

Table 18. Data Description

Variable	Description	Example
Users Types	Politician	Modi, Gandhi, Jayalalithaa, etc
	Parties	BJP, AAP, INC etc
	Media	The Times of India, The Hindu, etc
Tweets Types	Tweet	#SadacharYatra The #BJP government had promised employment to 13 lakh in 2007 but only 80,000 got jobs. #SadacharYatra
	Topic	job, government, lakh, get, bjp, political, provide, lose, claim, pass
Hashtags Types	Hashtags	#Narendra Modi For PM, #BJP, #ANI, etc
Sentiment Scores	Sentiment Scores	Number (From -1 to 1)

5.2. Experiment Setting

5.2.1. Baseline

We compare our model with the following methods. These model are random walk based model and graph neural network model. We compare these models' performance with our model's performance considering the fairness and the prediction:

DeepWalk: Perozzi et al (2014) built the DeepWalk to connect the word2vec method and graph embedding method. The first step is using the truncated random walks to transform the topology data into the sequential data. The second step is using the word2vec method to calculate the embedding to represent the users and items.

Node2vec: Grover and Leskovec (2016) built the Node2vec model to embed the node into the vector space. The first step is using the biased random walks to search each node in a local graph network. The second step is optimize the likelihood function for each node embeddings.

Metapath2vec: Dong et al (2017) built the Metapath2vec model to capture different types of information from a given network. The first step is building a meta-path to connect all different types of nodes. The second step is calculating the transition probability of each meta-path.

GCN: Kipf and Welling (2016) combined convolutional neural network and graph data together. The key idea is scanning the graph topology data by spectral filters. It helps to scan the convolution neural network in a graph data.

GAT: Veličković et al (2018) used attention mechanism from the computer vision field to apply on the graph structure data. This method used a shared attention weights to calculate each nodes embedding. It allows different importance for each node.

FairHIN: Zeng et al (2021) built a fair meta-path sampling method to capture the features around the heterogeneous nodes and edges. They used the demographic parity and the equal opportunity as measurement to capture the fairness loss.

FairGNN: Dai and Wang (2021) built the adversarial network to minimize the bias generated from the unfairness sampling. They also minimize the covariance constraint to stabilize the training process.

5.2.2. Parameter Setting

We construct based on the proposed model given in section 4.2. The optimization is based on the Adam algorithm implemented with PyTorch. The learning rate is set to {0.001, 0.005, 0.01,0.05}, the random walk length is {10, 50}, the window size is set to 5.

Our model parameters setting are as follows: batch size is 246, the embedding size of entity type is 32, the embedding size of edge is 64.

5.3. Performance Comparison

In our research setting, our classes are defined [-1 to -0.5) as class 1, [-0.5 to 0) as class 2, [0 to 0.5) as class 3, and [0.5 to 1] as class 4. We mainly choose the following four measurements to evaluate our model. The details of the formula is described in Appendix M.

Performance Measurement:

Recall Ratio (Recall@K), means how much percentage we truly predict the sentiment scores for each user given the ground truth sentiment scored samples based on the recommendation algorithm. We set the K=20. The higher Recall Ratio, the better of the predicted results on the sentiment scores given the recommendation algorithm. The details of calculating the Recall formula is described in appendix M.

Hit Ratio (HR@K), means how much percentage we truly predict the sentiment scores of the focal user based on the recommendation algorithm given these users' corresponding tweets in the testing sets. The details of calculating the Hit Ratio formula is described in appendix M.

$$\mathrm{HR@k} = \text{Number of Hits@}\frac{\mathrm{K}}{|\text{Testing Sets of Tweets}|}$$

We set K =20

Normalized Dicounted Cummulative Gain (NDCG@K), measures for each user, after we give the predicted sentiment score based on the recommendation algorithm. How these sentiment scores rank in the predicted result affect the result in testing set. We set K=20. The details of calculating the NDCG formula is described in appendix M.

AUC, measures the probability that how much percentage the truly recommended scores to the focal user compared with how much percentage this focal user is not truly recommended the correct scores. The higher AUC score, the better of the model to give the recommendation. The details of the AUC is described in appendix M.

From table 19 to table 22, we show the recommendation results about the nodes with 10,000, 20,000, and 50,000. Our model can give the best recommendation results in all different kinds of measurement.

In table 19, we compare our model with the benchmark models in different nodes size. The result shows that our model has the highest Recall ratio. It means our recommendation model gives the best truly predicted sentiment scores results given the ground truth sentiment scores samples.

Table 19. Recall@20 Results on three datasets. The best method is bolded

Method	10K node	20K node	50K node
DeepWalk	0.1142	0.1139	0.1104
Node2vec	0.1180	0.1164	0.1158
Metapath2vec	0.1195	0.1184	0.1143
GCN	0.1358	0.1341	0.1310
GAT	0.1389	0.1376	0.1354
DySAT	0.1794	0.1789	0.1745
DHNE	0.1784	0.1766	0.1727
Our model	**0.2371**	**0.2295**	**0.2289**

In table 20, we compare our model with the benchmark models in different nodes size. The result shows that our model has the highest Hit Ratio. It means our recommendation model gives the best truly predicted sentiment scores of the focal user given these users' corresponding tweets.

Table 20. HR@20 Results on three datasets. The best method is bolded

Method	10K node	20K node	50K node
DeepWalk	0.1632	0.1604	0.1582
Node2vec	0.1604	0.1693	0.1683
Metapath2vec	0.1659	0.1641	0.1633
GCN	0.1885	0.1889	0.1858
GAT	0.1897	0.1884	0.1852
DySAT	0.2302	0.2293	0.2286
DHNE	0.2286	0.2271	0.2257
Our model	**0.2877**	**0.2794**	**0.2742**

In table 21, we compare our model with the benchmark models in different nodes size. The result shows that our model has the highest NDCG value. It means our recommendation model gives the best truly predicted sentiment scores of the focal user given these users' corresponding tweets correlations' ranking.

Table 21. NDCG@20 Results on three datasets. The best method is bolded

Method	10K node	20K node	50K node
DeepWalk	0.0627	0.0623	0.0696
Node2vec	0.0612	0.0606	0.0689
Metapath2vec	0.0636	0.0627	0.0692
GCN	0.0882	0.0874	0.0848
GAT	0.0893	0.0886	0.0844
DySAT	0.1247	0.1236	0.1215

Table 21. (cont'd)

DHNE	0.1276	0.1253	0.1232
Our model	**0.1842**	**0.1831**	**0.1844**

In table 22, we compare our model with the benchmark models in different nodes size. The result shows that our model has the highest AUC value. It means our recommendation model gives the best truly recommended scores to the focal user compared with how much percentage this focal user is not truly recommended the correct scores.

Table 22. AUC Results on three datasets. The best method is bolded

Method	10K node	20K node	50K node
DeepWalk	0.7694	0.7628	0.7469
Node2vec	0.7921	0.7942	0.7836
Metapath2vec	0.8015	0.8153	0.8036
GCN	0.8153	0.8168	0.8045
GAT	0.8166	0.8173	0.8034
DySAT	0.8626	0.8621	0.8538
DHNE	0.8693	0.8658	0.8594
Our model	**0.8842**	**0.8856**	**0.8811**

5.4. Fairness of the recommendation

After we compared our model with the benchmark models with the performance of accuracy, we also need to compare our model with the benchmark models with the performance of fairness. There is no standard way to measure the fairness, we choose the demographical parity difference and the equal opportunity difference as the measurement to evaluate the fairness of the model.

5.4.1 Fairness Measures:

Demographical Parity (DP): The Statistical Parity means for each focal user, he or she got recommendation should be independent to his or her attributes (features). We create the difference of the Demographical Parity as the measurement

$$\text{Demographical Parity Difference} = P(\hat{y} = 1|x_{sensitive} = 0) - P(\hat{y} = 1|x_{sensitive} = 1)$$

The equal opportunity (EO): The equal opportunity means the probability of the focal user got assigned with a positive outcome should be equal for all other subgroup users. We create the difference of the Equal Opportunity as the measurement

$$\text{Equal Opportunity Difference} = P(\hat{y} = 1|x_{sensitive} = 0, y = 1) - P(\hat{y} = 1|x_{sensitive} = 1, y = 1)$$

In table 23, we compare our model with the random walk based models, the graph neural network based model with the demographic parity. Our model have the best predicted scores results given the sentiment attributes as the location.

Table 23. Demographic Parity Results on three datasets. The best method is bolded

Method	10K node	20K node	50K node
DeepWalk	0.1274	0.1249	0.1263
Node2vec	0.1294	0.1258	0.1303
Metapath2vec	0.1352	0.1395	0.1358
GCN	0.1742	0.1763	0.1798
GAT	0.1803	0.1865	0.1877
FairHIN	0.2048	0.2085	0.2094
FairGNN	0.2103	0.2162	0.2189
Our model	**0.2251**	**0.2263**	**0.2285**

In table 24, we compare our model with the random walk based models, the graph neural network based model with the equal opportunity. Our model have the best predicted scores

results given the corresponding ground truth sentiment scores and the sentiment attributes as the location.

Table 24. Equal Opportunity Results on three datasets. The best method is bolded

Method	10K node	20K node	50K node
DeepWalk	0.1301	0.1297	0.1307
Node2vec	0.1316	0.1302	0.1341
Metapath2vec	0.1348	0.1363	0.1396
GCN	0.1854	0.1821	0.1863
GAT	0.1865	0.1879	0.1896
FairHIN	0.2143	0.2174	0.2196
FairGNN	0.2269	0.2286	0.2293
Our model	**0.2301**	**0.2317**	**0.2321**

6. CONCLUSION

The recommendation algorithm in social network is very important for the politician to diffuse their political opinions. Compared with traditional survey method, electronic election is more efficient and popular for their followers. Therefore, using the twitter as campaign tool to declare the political polices and attract more voters is very important to the political candidates.

In order to hear the voice of the minority of users, platform need to design the fairness recommendation algorithms to recognize the preference of the users. However, all of the recommendation models are would give the bias results because there is always sampling bias happened during the training process. To solve this bias in the post-processing part is what our proposed model work.

In this paper, we proposed a novel meta-path based recommendation method on the election social network. This model learns the information from the different types of nodes, edges, and time periods. Our model consider both the prediction performance and the fairness of the model. In order to calculate the prediction of the recommendation, we use the advanced meta-path based model to capture the heterogeneous information of the network. In the meantime, we combine both meta-path based embedding method with the attention based method. This can somewhat help to decrease the difference between the users belong to the majority group and the users belong to the minority group.

The experimental results shows that our model performs better than baselines models on the real world dataset as the performance of Recall ratio, AUC, Hit Ratio, and NDCG. Our model also performs better than baselines models on different kind of dataset as the performance of demographic parity and the equal opportunity.

APPENDICES

APPENDIX A. Notation in essay one

Table A.1. Equal Opportunity Results on three datasets. The best method is bolded

Notation	Description
x_i	Demographic features for user i
y_i	Potential outcome for user i after get treatment t
$y_i(0)$	Counterfactual outcome for user i after get treatment $t = 0$
$y_i(1)$	Factual outcome for user i after get treatment $t = 1$
t_i	User i get treatment t
A	Adjacency matrix of the observed network G
$G\{E, V\}$	We have edge E and vertex V to represent a graph G
μ and ν	Any two node μ and ν select from graph G
z	The state layer in Graph Neural Network
W_k	Weight matrix in Graph Neural Network on k^{th} iteration
B_k	Bias matrix in Graph Neural Network on k^{th} iteration
N	The number of neighbor nodes
X	Demographic features Space
Y	Outcome Space
H	Hidden Space
ϕ	The embedding, also called encoder function to map the tuple ($\{x_i, y_i, t_i\}$, A) into the hidden space h
ψ	The decoder function to map tuple ($\{x_i, y_i, t_i\}$, A) into the outcome space
h	The hidden space, also called embedding space h

APPENDIX B. Propensity Score, Inverse Probability Treatment Weighting estimation, and Doubly Robust method

Un-confound means:

$$\mathrm{y_i}(0), \mathrm{y_i}(1) \perp \mathrm{t_i} \mid x_i, c_i$$

We can calculate the Propensity Score as : $\mathrm{e}(\mathrm{x_i}) = \mathrm{P}(\mathrm{T_i} = 1 | \mathrm{X_i} = \mathrm{x_i}) \ \forall \mathrm{x} \in \mathrm{X}$

If the propensity score is a good balancing score, we will get:

$$\mathrm{y_i}(0), \mathrm{y_i}(1) \perp \mathrm{t_i} \mid e(x_i)$$

This way is very efficient to remove the biases coming from the non random assignment.

Under confound environment (also called balanced property)

$$\mathrm{e}(\mathrm{x_i}) = \mathrm{E}[\mathrm{T_i} = 1 | e(x_i)]$$

Inverse Probability Treatment Weighting estimation is the way to balance the difference between two groups:

$$\tau_{\mathrm{IPTW}} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{W_i Y_i}{\hat{e}(x_i)} \right) - \left(\frac{(1 - W_i) Y_i}{1 - \hat{e}(x_i)} \right)$$

The quality of this estimator depends on the estimation of $\hat{e}(x_i)$

Because there exists confound, we cannot get good propensity scores.

Covariate Balanced Propensity Score is the most advanced tool to handle the network data. It is also the double robust method.

Doubly Robust method is a good estimator when one of the two assumptions are right:

1 . Outcome function is linear but propensity is not logistic

2 . Outcome function is not linear but propensity is logistic

Because the outcome function $\mathrm{Y} = \phi(\mathrm{x})$ is nonlinear, and propensity score cannot always achieve by logistic regression, DR method is still not good enough.

Our aim is to minimize the balanced score and variance, which is the equal to minimize the IPM + regulator form. The math proof can be found in Kallus (2020)

$$E[(\tau_w - \tau)^2 | X, T] = \frac{1}{n^2} \sum w_i f(x_i)^2 + \frac{1}{n^2} \sum w_i^2 \sigma^2 x_i$$

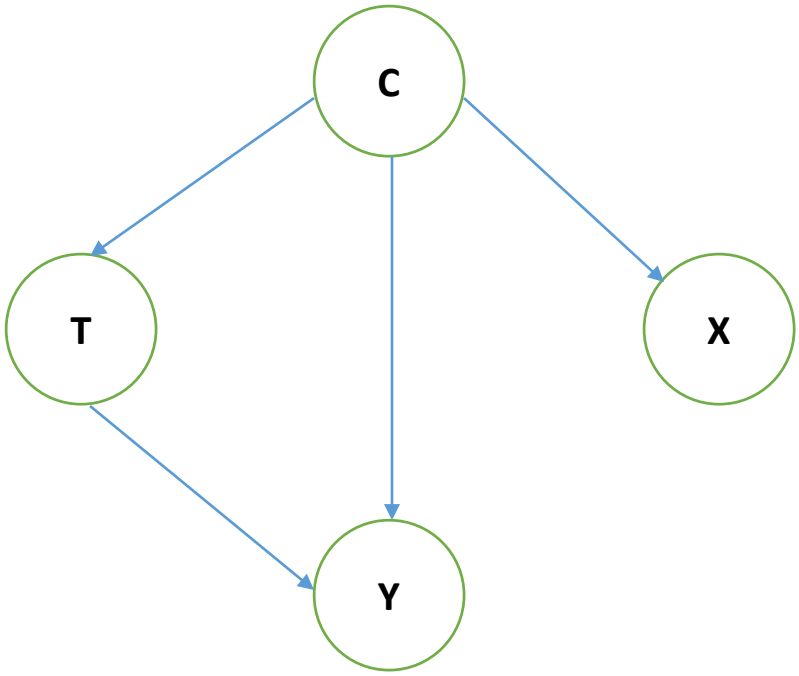
APPENDIX C. Ignore the hidden confounders

Go back to review the content of Assumption 1 : There is no hidden confounder C affect on both X, Y and T

$$y_i(0), y_i(1) \perp t_i \mid x_i, c_i$$

Pearl (2000) proposed the causal graph model to identify the causal inference problem. The basic idea is finding an observed proxy variable to approximate the unobserved confounder C. In this setting, we need to approximate the C by the tuple (X,Y.T)

Figure A.1. Causal Graph Model to identify the causal inference problem



Because we cannot measure the C part, directly calculate the conditional probability $P(Y|X,T)$ may generate the estimation bias. Pearl gave a method called do-calculus to solve this problem. Do-Calculus can be nominated by $P(Y|X, do(T))$ and transformed to the conditional probability as follows

$$\mathrm{P}\big(\mathrm{Y}=\mathrm{y}\big|\mathrm{X},\mathrm{do}(\mathrm{T}=\mathrm{t})\big)=\sum_{C\in[0,1]}P_{Y|T,C}\left(y|t,c\right)P_C(c)$$

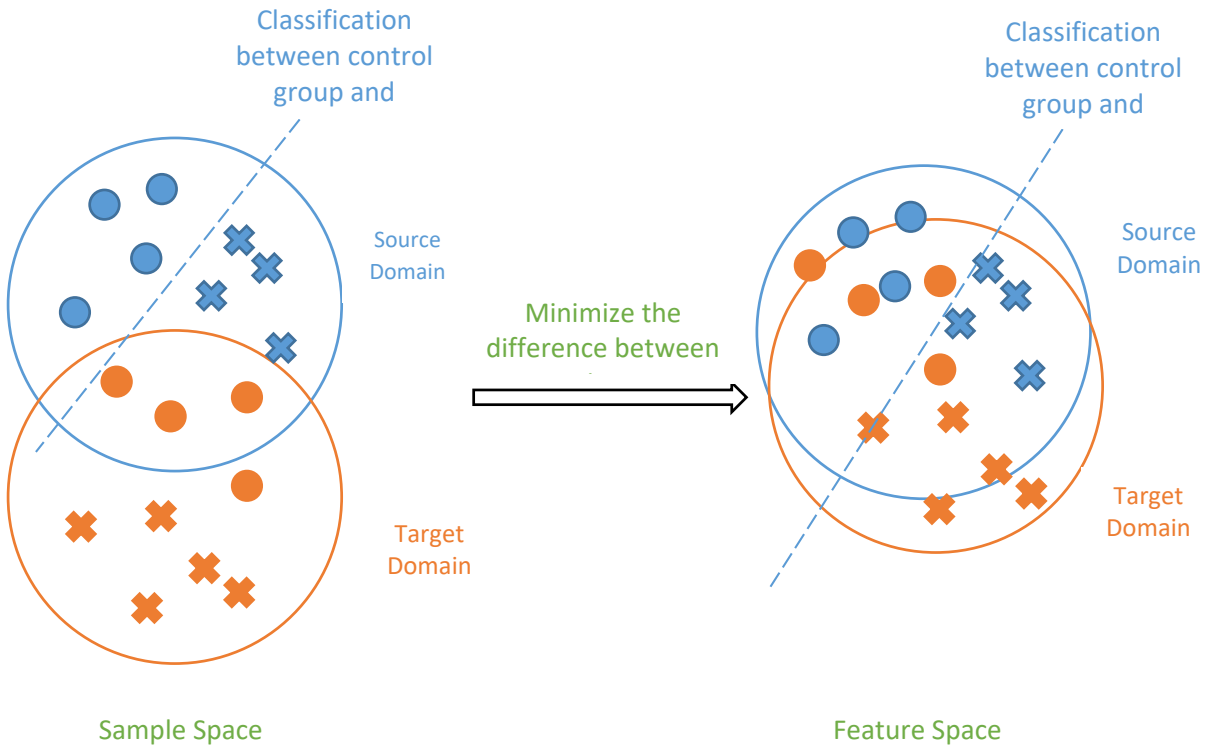
In our average treatment effect setting, we can calculate the ATE as

$$\begin{aligned}\mathrm{ATE}&=\mathrm{P}\big(\mathrm{Y}=1\big|\mathrm{X},\mathrm{do}(\mathrm{T}=1)\big)-\mathrm{P}\big(\mathrm{Y}=1\big|\mathrm{X},\mathrm{do}(\mathrm{T}=0)\big)\\&=\sum_{C\in[0,1]}\left(P_{Y|T,C}(1|1,c)-P_{Y|T,C}(1|0,c)\right)P_C(c)\end{aligned}$$

APPENDIX D. Diagram of the Domain Adaptation for counterfactual framework

Let us have a look at the following diagram, circles mean the treated group and dots mean the control group. We need to use Maximum Mean Distance or Wasserstein 1 Distance to calculate the distance between two data domains. If these two data domains can get closer, the covariates variables become more balanced on the representation space. The dashed line is the mean square error to measure the error between predict y and real y

Figure A.2. Domain Adaptation for counterfactual framework



APPENDIX E. Maximum mean discrepancy (MMD)

The most common way to measure the distance between the target data and source data is the Maximum mean discrepancy (MMD). Let define the source data x is coming from a distribution p and the target data y is coming from a distribution q

First step, define a function f mapping the sample into a hidden space. $f(x)\ and\ f(y)$. The mean value between these two function is $E(f(x))$ and $E(f(x))$

Second step, calculate their difference and find the maximum function f

$$\mathrm{MMD}[\mathcal{F}, \mathrm{p}, \mathrm{q}] = \sup_{f \in \mathcal{F}} \left(E\left(f_{x \sim p}(x)\right) - E\left(f_{y \sim q}(x)\right) \right)$$

Third step, replace the distribution with the sample

$$\mathrm{MMD}[\mathcal{F}, \mathrm{x}, \mathrm{y}] = \sup_{f \in \mathcal{F}} \left(\frac{1}{m} \sum_{i=1}^{m} f(x_i) - \frac{1}{n} \sum_{i=1}^{n} f(y_i) \right)$$

If the distribution of x is identical to the distribution of y, their expectation after the mapping should be the same. Therefore, if we want to transfer the source domain to the target domain successfully, we just need to minimize the maximum mean discrepancy value.

Sometimes, the function f is not easy to calculate, we need to construct a reproduction kernel Hilbert space (RKHS) to make this adaptation domain process easy to compute.

$$\begin{aligned}
\mathrm{MMD}[\mathcal{F}, \mathrm{p}, \mathrm{q}] &= \sup_{f \in \mathcal{F}_H \leq 1} \left(E\left(f_{x \sim p}(x)\right) - E\left(f_{y \sim q}(x)\right) \right) \\
&= \sup_{f \in \mathcal{F}_H \leq 1} \left(E_p[\langle \phi(x), f \rangle_H] - E_q[\langle \phi(x), f \rangle_H] \right) \\
&= \sup_{f \in \mathcal{F}_H \leq 1} \left(\langle \mu_p - \mu_q, f \rangle \right) \\
&= \left\| \mu_p - \mu_q \right\|_H
\end{aligned}$$

$$\mathrm{MMD}^2[\mathcal{F}, \mathrm{p}, \mathrm{q}] = \left\| \mu_p - \mu_q \right\|_H^2 = E_p \langle \phi(x), \phi'(x) \rangle_H + E_p \langle \phi(y), \phi'(y) \rangle_H - 2E_{p,q} \langle \phi(x), \phi(y) \rangle_H$$

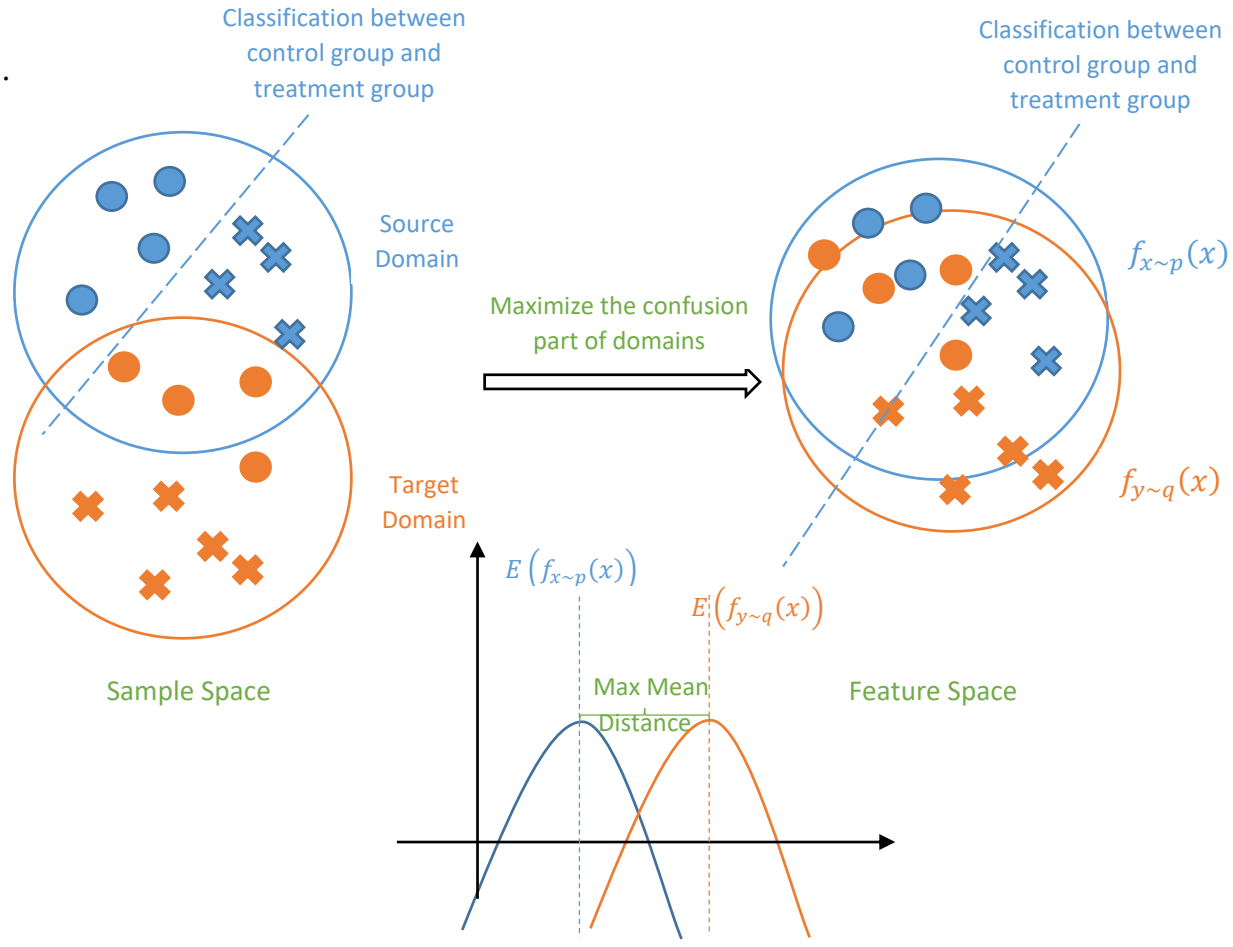
If we choose the Gaussian kernel

$$\mathrm{MMD}^2[\mathcal{F}, \mathrm{p}, \mathrm{q}] = \frac{1}{m(m-1)}\sum_{i\neq j}^{m} k(x_i, x_j) + \frac{1}{n(n-1)}\sum_{i\neq j}^{n} k(y, y_j) - \frac{2}{mn}\sum_{i\neq j}^{m,n} k(x_i, y_j)$$

Where $k(x, x') = exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)$

Figure A.3. Domain Adaptation for counterfactual framework with Maximum mean discrepancy

.



APPENDIX F. Wasserstein Distance

Another way to calculate the distance between two distributions is Wasserstein Distance. Compare with the MMD and some other distance measure such as KL divergence, JS divergence, Wasserstein Distance has some advantages :

1 . It can measure the distance between discrete distribution and continuous distribution

2 . Even two distributions has very long distance, the Wasserstein Distance can still measure the length but not give constant or zero result

3 . Wasserstein barycenter can describe the geographic character compare to the Euclidean average

However, Wasserstein Distance is very hard to compute, only Wasserstein 1 Distance can be calculated easily. The basic idea is treating the distribution as a set of stones. If we take each stone from one distribution p to another distribution q, we want to find a minimum path to achieve the task. After we find the minimum path for each stone and sum all these path up, we can equally get the distance between two distributions.

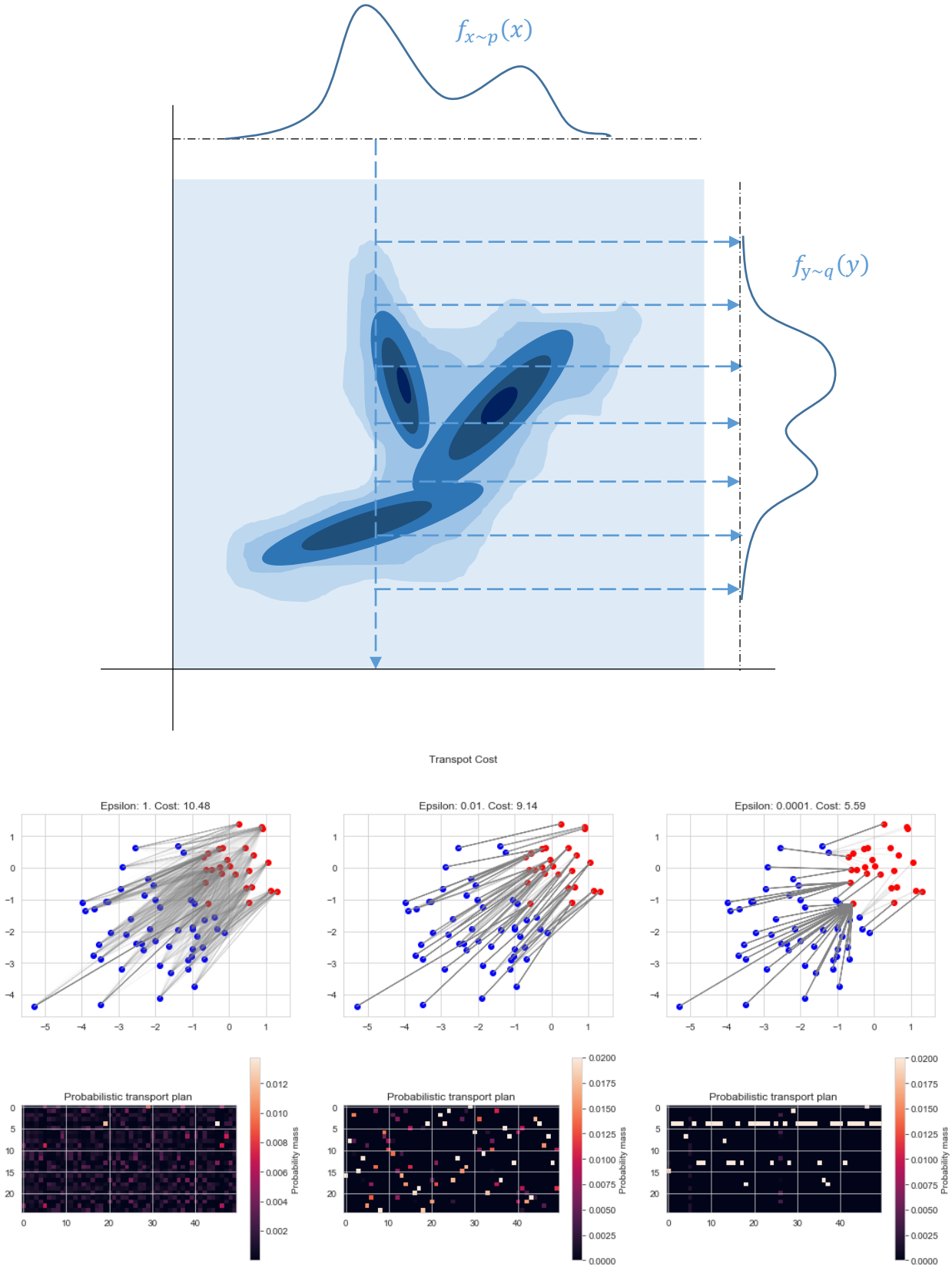
$$W_p(\mu,\nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int \|x-y\|^p d\gamma(x,y) \right)^{\frac{1}{p}}$$

where p is the lipschitz constraint, μ, ν are two probability measures, and $\gamma(x,y)$ is the joint distribution

If we choose dimension equals to 1, it becomes the Wasserstein 1 Distance:

$$W_p(\mu,\nu) = \left(\int_0^1 \|x-y\|^p d\gamma(x,y) \right)^{\frac{1}{p}} = \left(\sum_{i=1}^{n} \|x_i - y_i\|^p \right)^{\frac{1}{p}}$$

Figure A.4. Wasserstein Distance



APPENDIX G. Node2Vec and :Latent Features

Graph embedding extends the representation from structure data to graph data. Given a graph $G\{E, V\}$, we have edge E and vertex (node) V to represent a graph. We need to find a mapping function ϕ to encode each vertex V into a hidden embedding space $\mathcal{H}$. This embedding space is the same with the representation space. We expect that embedding space can still contain the information in sample space. In another word, the similarity between node μ and ν should be as equal to the similarity between embedding node $\phi(\mu)$ and $\phi(\nu)$ as possible

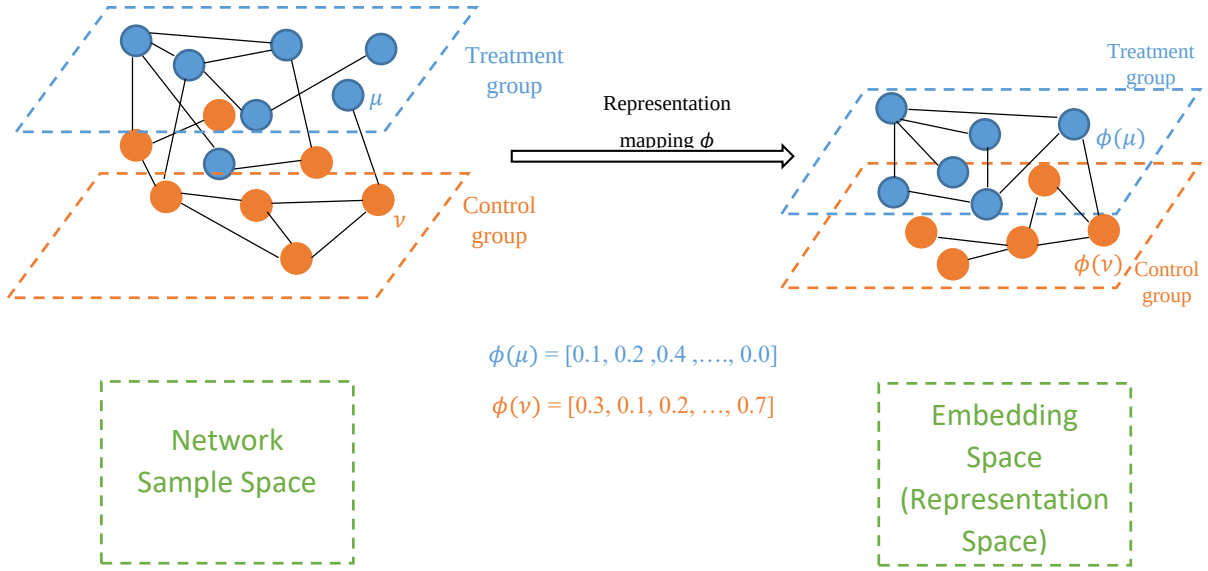
$$Similarity\ (\mu, \nu) \approx \phi(\mu)^T \phi(\nu) \tag{5}$$

For each node, we use its feature, con-current probability, and etc, to represent the vector. For example, node μ can be vectorized to $\phi(\mu) = [0.1,0.2,0.4, \ldots,0.0]$ and node ν can be vectorized to $\phi(\nu) = [0.3,0.1,0.2, \ldots,0.7]$. Each node is encoded as 1*D dimension vector and all nodes are encoded as V*D dimensions matrix. After we input the graph G, label Y and calculate the object function

$$L = \sum_{\mu,\nu \in V*V} \| Similarity\ (\mu, \nu) - \phi(\mu)^T \phi(\nu) \|^2 \tag{6}$$

, we can get the embedding space. The encoder is just a simple lookup style embedding and the similarity function can be chosen from adjacency based, multi-hop based, and Random-walk based methods.

Figure A.5. Node2Vec and Latent Features



Edge relationship is always represented by an adjacency matrix A, we use it to replace the similarity function $Similarity\ (\mu, v)$. The similarity function can be adjacency based, multi-hop based, and Random-walk based. The simple encoder function ϕ is just a look-up function, we will replace it by the graph neural network later.

Table A.2. Pros and Cons of the Similarity function Choices

Embedding Method	Loss Function	Pros and Cons
Adjacency based	$L = \sum_{\mu,v \in V*V} \left\| A_{\mu,v} - \phi(\mu)^T \phi(v) \right\|^2$	Only consider the complete graph
Multi-hop based	$L = \sum_{\mu,v \in V*V} \left\| A^k_{\mu,v} - \phi(\mu)^T \phi(v) \right\|^2$	Need to define the pairwise node similarity by researcher
Random-walk based	$L = \sum_{\mu \in V} \sum_{\in N_R(v)} - log(P(v \mid \phi(\mu)))$	It can be any random walk chain to generate the pairs nodes.

We choose the Random-walk based method because:

1) It can define the relationship very flexible, only consider the pairs node that con-current on the random-walk chain

2) Compare with two other methods, it is efficient to compute.

The Random-walk embedding method is using the skip-gram idea from the word to vector method. The basic idea is as follows:

Step1: Choosing any node μ and generating the random walk start from the node μ with some strategy S.

Step2: Calculate the probability of visiting a node ν from the node μ, $P(\nu|\phi(\mu))$, by the random-walk chain

Step3: Optimize the loss function $L = \sum_{\mu \in V} \sum_{\nu \in N_S(\nu)} -log(P(\nu|\phi(\mu)))$. Plug in the softmax function, we can get the following function

$$L = \sum_{\mu \in V} \sum_{\nu \in N_R(\nu)} -log\left(\frac{\exp(\phi_\mu^T \phi_\nu)}{\sum_{n \in V} \exp(\phi_\mu^T \phi_\nu)}\right) \quad (7)$$

Step4: Negative Sampling for compute efficient

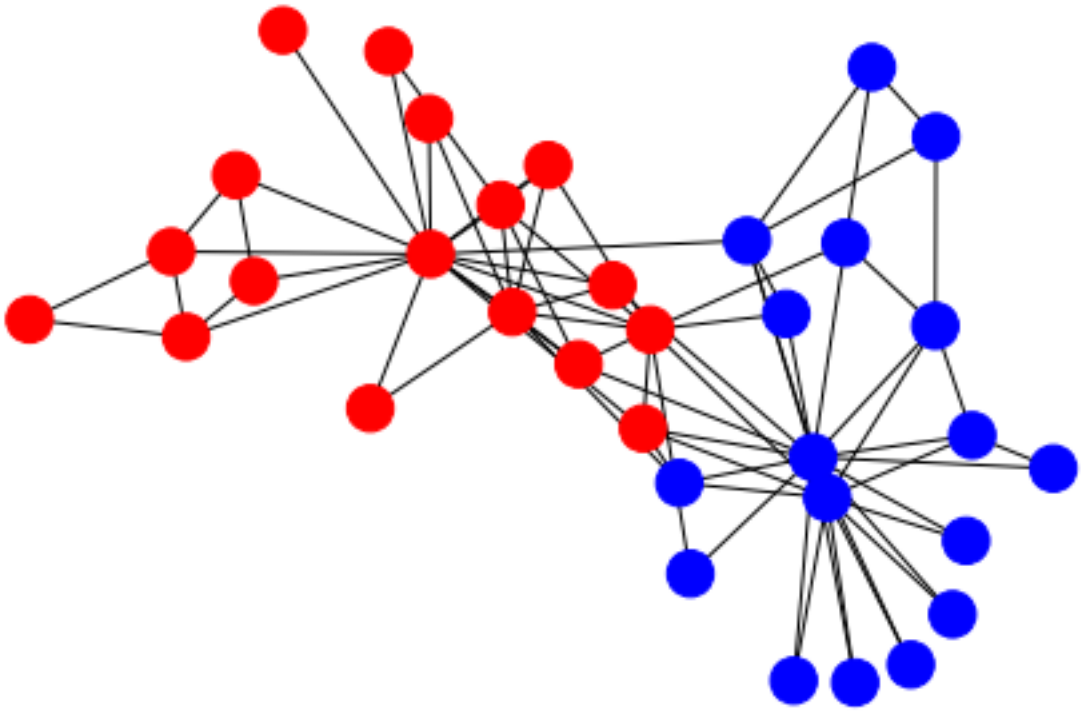
$$log\left(\frac{\exp(\phi_\mu^T \phi_\nu)}{\sum_{n \in V} \exp(\phi_\mu^T \phi_\nu)}\right) = \log\left(\sigma\left(\phi_\mu^T \phi_\nu\right)\right) - \sum_{n \in V} \log \sigma(\phi_\mu^T \phi_\nu) \quad (8)$$

There is an example to illustrate this process, let's assume we have a small network and try to embed it into the vector space by skip-gram[9]

[9] The skip-gram method can be used to solve the word2vec problem The basic form is $\frac{\log((w,c)D)}{b\,(w)(c)}$, where w is the word and c is the context, and D is the total number of word-context pairs

What kind of latent features we can extract from nodes? Let assume we have a network as follows:

Figure A.6. Node2Vec Visualization



We have 34 nodes, and 78 edges. Red and Blue are just label for each nodes, let's say democratic and republican, we want to extract some features to classify each node

By using random walk method, we can generate walk path on each node, choosing walk length as 10, it generates 34*10 walks:

[0, 11, 0, 11, 0, 12, 3, 12, 3, 1]

[0, 21, 0, 2, 7, 2, 0, 8, 2, 8]

[0, 2, 3, 7, 0, 21, 0, 6, 0, 31]

[0, 4, 0, 5, 10, 0, 13, 2, 7, 2]

[0, 31, 0, 31, 0, 17, 0, 8, 33, 9]

[0, 13, 0, 2, 27, 2, 3, 7, 3, 1]

[0, 1, 17, 1, 0, 3, 1, 3, 2, 1]

[0, 13, 2, 1, 2, 0, 2, 1, 2, 1]

[0, 12, 0, 31, 0, 10, 5, 16, 5, 6]

[0, 31, 28, 31, 28, 33, 28, 33, 9, 33]

[1, 13, 1, 2, 1, 17, 1, 0, 13, 0]

Put these walk paths into the skip-gram model, we can get the features on each node, we choose hidden space dimension as 12 :

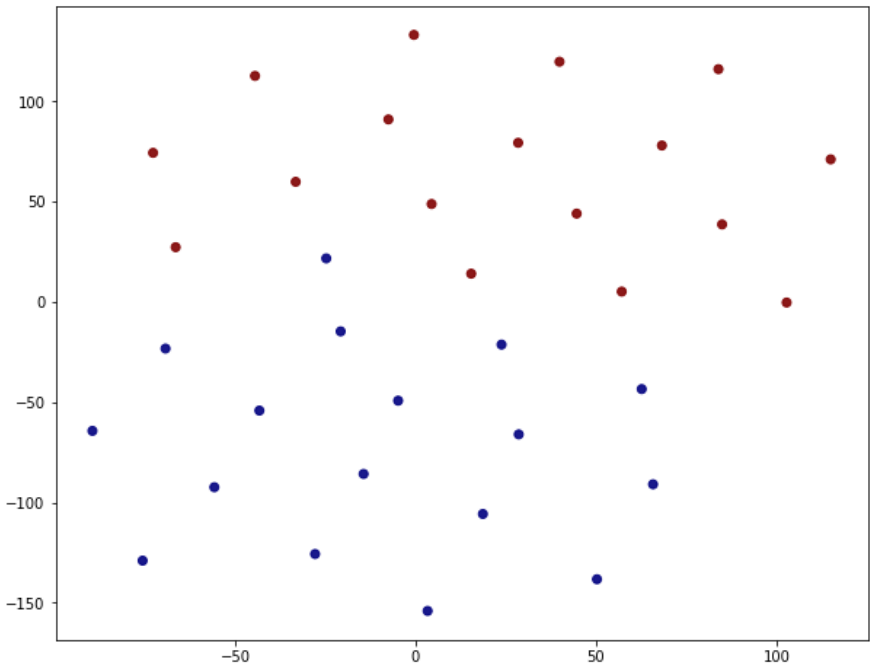
Node 1 has features with 1*12:

array([0.03700204, -0.0196846 , 0.038861 , 0.00561627, 0.02323475,
    -0.04270516, 0.01880202, 0.00162641, -0.02773627, -0.02885933,
    -0.02692292, -0.02374379], dtype=float32)

The element means the probability of each node happened on each walk path. By using these features, we can reconstruct the node location and graph structure in the R dimension space. The node embeddings is 34*12 matrix

After this encoding step, we need to reconstruct these features on a 2D space figure:

Figure A.7. Node2Vec for Node Classification



We can find that our 34*12 features space can nicely help to classify each node.

APPENDIX H. Balanced Covariates Measures and Imbalanced Covariates Measures

From Table A.3 to Table A.4 show the balanced testing results of covariates such as retweet_count, scource, followers_count, location, statuses_count, friends_count, and favourites_count. The standard mean value and the empirical cumulative density function are expected to close to zero and the difference Variance Ratio is expected to close to one.

Table A.3. Balanced Covariates Measures on two assigned group

Summary of Balance for Matched Data:	Means Treated	Means Control	Std. Mean	Diff. Var. Ratio	eCDF Mean	eCDF Max
distance	0.5528	0.5525	0.0022	1.0210	0.0024	0.0222
retweet_count	62.0180	41.3845	0.1830	0.4142	0.0664	0.1531
source	2.7947	2.6499	0.0406	0.6929	0.0256	0.1489
followers_count	989.6921	1208.5635	-0.0480	0.7915	0.0480	0.1187
location	80.6824	89.9013	-0.0912	0.9353	0.0279	0.0901
statuses_count	10693.3301	13188.3687	-0.1118	0.7131	0.0420	0.1003
friends_count	423.0513	522.9355	-0.1642	0.6954	0.0472	0.0884
favourites_count	1154.4508	1525.9780	-0.1007	0.5520	0.0245	0.0578

Table A.4. Imbalanced Covariates Measures on two assigned group

Summary of Balance for Matched Data:	Means Treated	Means Control	Std. Mean	Diff. Var. Ratio	eCDF Mean	eCDF Max
distance	0.5528	0.4749	0.6384	0.7763	0.1537	0.2460
retweet_count	62.0180	149.0133	-0.7714	0.0691	0.0476	0.1003
source	2.7947	3.7570	-0.2696	0.5307	0.0370	0.1301
followers_count	989.6921	648.8513	0.0747	2.7630	0.0311	0.0743
location	80.6824	61.4963	0.1898	1.4801	0.0569	0.1295
statuses_count	10693.3301	7948.9352	0.1229	2.3092	0.0364	0.0668
friends_count	423.0513	425.7408	-0.0044	0.8257	0.0168	0.0546
favourites_count	1154.4508	1196.1753	-0.0113	0.7098	0.0261	0.0865

The Table A.5 and A.6 show the balanced testing results of observed covariates incorporating with the latent graph features from fea0 to fea19 generated from the GNN.

Table A.5. Balanced Covariates Measures on two assigned group with latent features

Summary of Balance for All Data:	Means Treated	Means Control	Std. Mean	Diff. Var. Ratio	eCDF Mean	eCDF Max
distance	0.5816	0.4443	0.8264	0.9102	0.2093	0.3116
retweet_count	22.9307	75.0648	-0.9012	0.0629	0.0546	0.1308
source	62.0180	149.0133	-0.7714	0.0691	0.0476	0.1003
followers_count	2.7947	3.7570	-0.2696	0.5307	0.0370	0.1301
location	989.6921	648.8513	0.0747	2.7630	0.0311	0.0743
statuses_count	80.6824	61.4963	0.1898	1.4801	0.0569	0.1295
friends_count	10693.3301	7948.9352	0.1229	2.3092	0.0364	0.0668
favourites_count	423.0513	425.7408	-0.0044	0.8257	0.0168	0.0546
fea0	0.1446	0.1437	0.0007	1.2589	0.0110	0.0386
fea1	-0.0729	-0.0264	-0.0411	0.9343	0.0113	0.0345
fea2	-0.1827	-0.2136	0.0246	0.9488	0.0177	0.0551
fea3	-0.1932	-0.1733	-0.0165	1.0388	0.0130	0.0361
fea4	-0.1254	-0.0065	-0.1092	0.8355	0.0367	0.0742
fea5	0.2121	0.1933	0.0151	1.0720	0.0081	0.0268
fea6	0.1856	0.1046	0.0728	0.9165	0.0110	0.0368
fea7	-0.2394	-0.2510	0.0092	1.0560	0.0124	0.0380
fea8	-0.2488	-0.1823	-0.0602	0.8491	0.0121	0.0433
fea9	0.0744	0.0662	0.0066	1.0174	0.0091	0.0302
fea10	0.3296	0.2738	0.0417	0.8545	0.0131	0.0357
fea11	-0.0661	0.0476	-0.0948	1.0436	0.0134	0.0374
fea12	0.0686	0.0456	0.0177	1.1773	0.0102	0.0278
fea13	0.0410	0.0742	-0.0299	0.9019	0.0253	0.0597
fea14	0.0189	0.0353	-0.0129	1.3130	0.0080	0.0282
fea15	0.0575	0.1055	-0.0422	0.8429	0.0181	0.0614

Table A.5. (cont'd)

fea16	0.1466	0.0688	0.0710	0.9113	0.0210	0.0544
fea17	0.0190	0.0857	-0.0612	0.6547	0.0213	0.0395
fea18	0.0129	0.0615	-0.0402	1.2819	0.0208	0.0483
fea19	0.0423	0.1656	-0.0867	0.8996	0.0093	0.0250

Table A.6. Imbalanced Covariates Measures on two assigned group with latent features

Summary of Balance for Matched Data:	Means Treated	Means Control	Std. Mean	Diff. Var. Ratio	eCDF Mean	eCDF Max
distance	0.5816	0.5808	0.0045	1.0166	0.0018	0.0194
favorite_count	22.9307	13.2903	0.1667	0.4832	0.0760	0.1993
retweet_count	62.0180	34.4026	0.2449	0.5106	0.0759	0.1548
source	2.7947	2.5350	0.0728	0.9014	0.0241	0.1370
followers_count	989.6921	1723.3125	-0.1608	0.6938	0.0416	0.1151
location	80.6824	89.2738	-0.0850	0.8979	0.0268	0.0717
statuses_count	10693.3301	14157.3344	-0.1552	0.6561	0.0368	0.0934
friends_count	423.0513	440.6401	-0.0289	0.8804	0.0229	0.0926
favourites_count	1154.4508	1396.1416	-0.0655	0.5590	0.0170	0.0456
fea0	0.1446	0.2248	-0.0594	1.9734	0.0521	0.1081
fea1	-0.0729	-0.0382	-0.0307	1.7857	0.0169	0.0463
fea2	-0.1827	-0.1964	0.0109	1.2429	0.0199	0.0535
fea3	-0.1932	-0.1918	-0.0012	1.5741	0.0178	0.0608
fea4	-0.1254	-0.1292	0.0035	1.4373	0.0215	0.0686
fea5	0.2121	0.2515	-0.0317	1.1118	0.0194	0.0618
fea6	0.1856	0.2147	-0.0261	1.0228	0.0369	0.0940
fea7	-0.2394	-0.2753	0.0286	1.4045	0.0242	0.0614
fea8	-0.2488	-0.2463	-0.0023	1.4477	0.0243	0.0605
fea9	0.0744	-0.0087	0.0670	1.1829	0.0278	0.0956
fea10	0.3296	0.2961	0.0250	1.1525	0.0137	0.0493
fea11	-0.0661	-0.0713	0.0043	1.5688	0.0232	0.0654

Table A.6. (cont'd)

fea12	0.0686	0.1819	-0.0874	1.3711	0.0312	0.0809
fea13	0.0410	0.1026	-0.0554	0.9797	0.0356	0.0910
fea14	0.0189	-0.0081	0.0214	1.8832	0.0241	0.0616
fea15	0.0575	0.1408	-0.0732	1.0851	0.0411	0.1004
fea16	0.1466	0.2004	-0.0490	1.3431	0.0503	0.1145
fea17	0.0190	-0.0175	0.0335	0.7556	0.0283	0.0855
fea18	0.0129	0.0718	-0.0487	1.5696	0.0243	0.0698
fea19	0.0423	0.0040	0.0269	1.0470	0.0291	0.0723

Table from A.7 to A.11 show the balanced testing result specifically on the propensity score with different network scale. (10,000 users, 20,000 users, 40,000 users, 80,000 users, and 100,000 users.)

Table A.7. Propensity Score Balanced Improvement (10k)

	Std. Mean	Diff. Var. Ratio	eCDF Mean
Original Imbalanced Dataset	0.6384	0.7763	0.1537
After Nearest Neighbor Matching	**0.0022**	1.0210	0.0024
Imbalanced Dataset including GNN embedding	0.8264	0.9102	0.2093
After Nearest Neighbor Matching (including GNN embedding)	0.0045	**1.0166**	**0.0018**

Table A.8. Propensity Score Balanced Improvement (20k)

	Std. Mean	Diff. Var. Ratio	eCDF Mean
Original Imbalanced Dataset	0.2220	0.3798	0.0272
After Nearest Neighbor Matching	0.0058	0.9895	0.0062
Imbalanced Dataset including GNN embedding	0.2945	0.7250	0.0696
After Nearest Neighbor Matching (including GNN embedding)	**0.0000**	**0.9982**	**0.0009**

Table A.9. Propensity Score Balanced Improvement (40k)

	Std. Mean	Diff. Var. Ratio	eCDF Mean
Original Imbalanced Dataset	0.2822	0.3654	0.0392
After Nearest Neighbor Matching	0.0131	1.0597	0.0008
Imbalanced Dataset including GNN embedding	0.3006	0.5173	0.0575

Table A.9. (cont'd)

After Nearest Neighbor Matching (including GNN embedding)	**0.0094**	**0.9608**	**0.0002**

Table A.10. Propensity Score Balanced Improvement (80k)

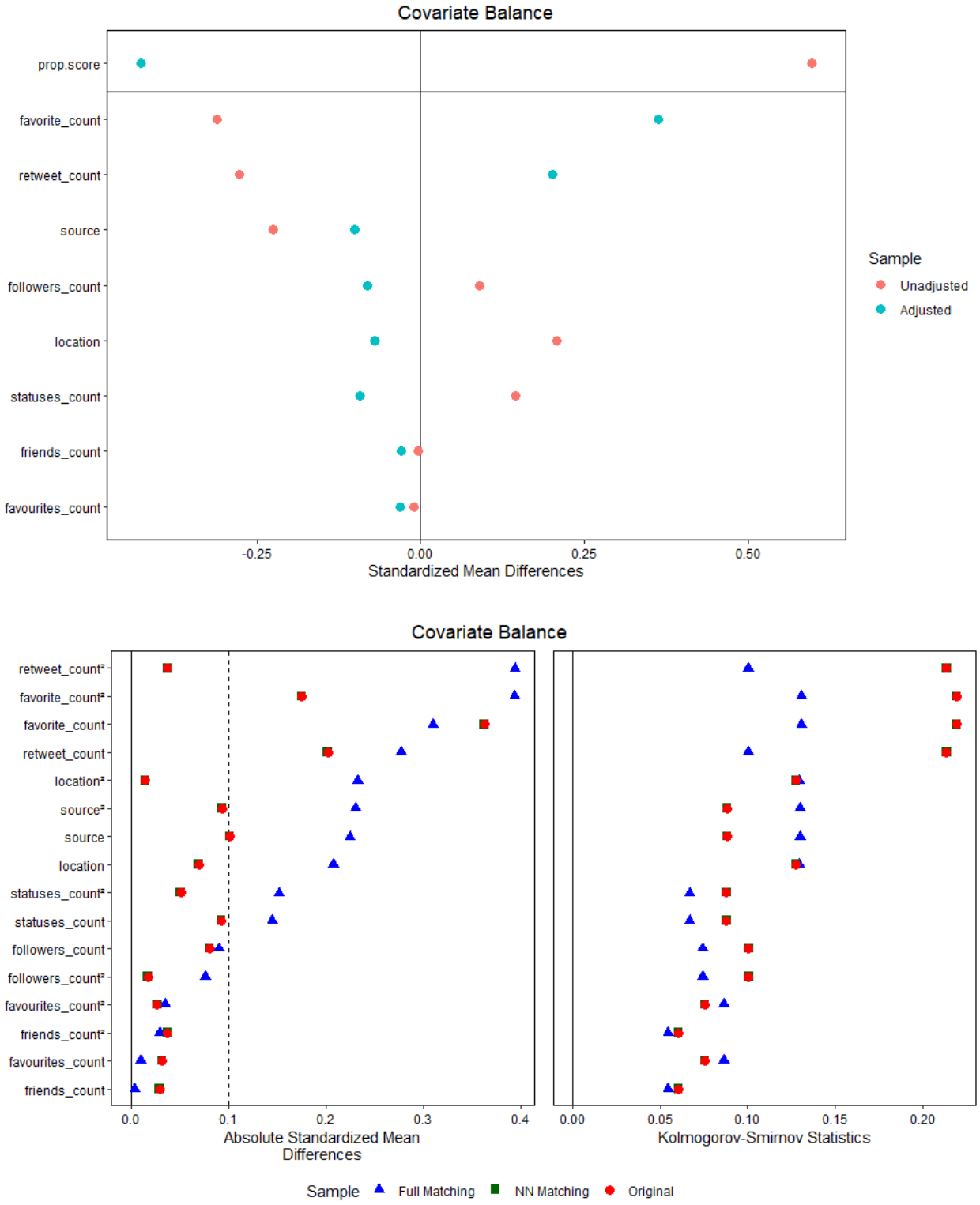
	Std. Mean	Diff. Var. Ratio	eCDF Mean
Original Imbalanced Dataset	0.2727	0.3801	0.0369
After Nearest Neighbor Matching	**0.0010**	0.9794	0.0003
Imbalanced Dataset including GNN embedding	0.2877	0.5101	0.0560
After Nearest Neighbor Matching (including GNN embedding)	0.0046	**0.9621**	**0.0001**

Table A.11. Propensity Score Balanced Improvement (100k)

	Std. Mean	Diff. Var. Ratio	eCDF Mean
Original Imbalanced Dataset	0.3845	0.3994	0.0658
After Nearest Neighbor Matching	0.0004	1.0022	0.0001
Imbalanced Dataset including GNN embedding	0.5915	0.6630	0.1451
After Nearest Neighbor Matching (including GNN embedding)	**0.0002**	**1.0009**	**0.0000**

The table A.11 gives the visualization of the covariates differences between treated group and the control group. Two figures show that after the balanced, (based on the nearest neighbor matching) the covariates differences between treated group and the control group becomes smaller than before the balanced. Table A.12 shows the visualization of the covariates differences including the latent graph features between the treated group and the control group. The differences become smaller after the balanced.

Figure A.8. Covariate Balanced Table for theory based features



The Table A-8.7 shows the visualization of the covariates distribution differences between treated group and the control group. Normally, after the balanced, their differences become smaller than before. The Table A-8.9 and the Table A-8.10 show the visualization of the covariates distribution differences including the latent graph features between the treated group and the control group. The differences become smaller after the balanced.

Figure A.9. Distribution Balance for each theory based feature

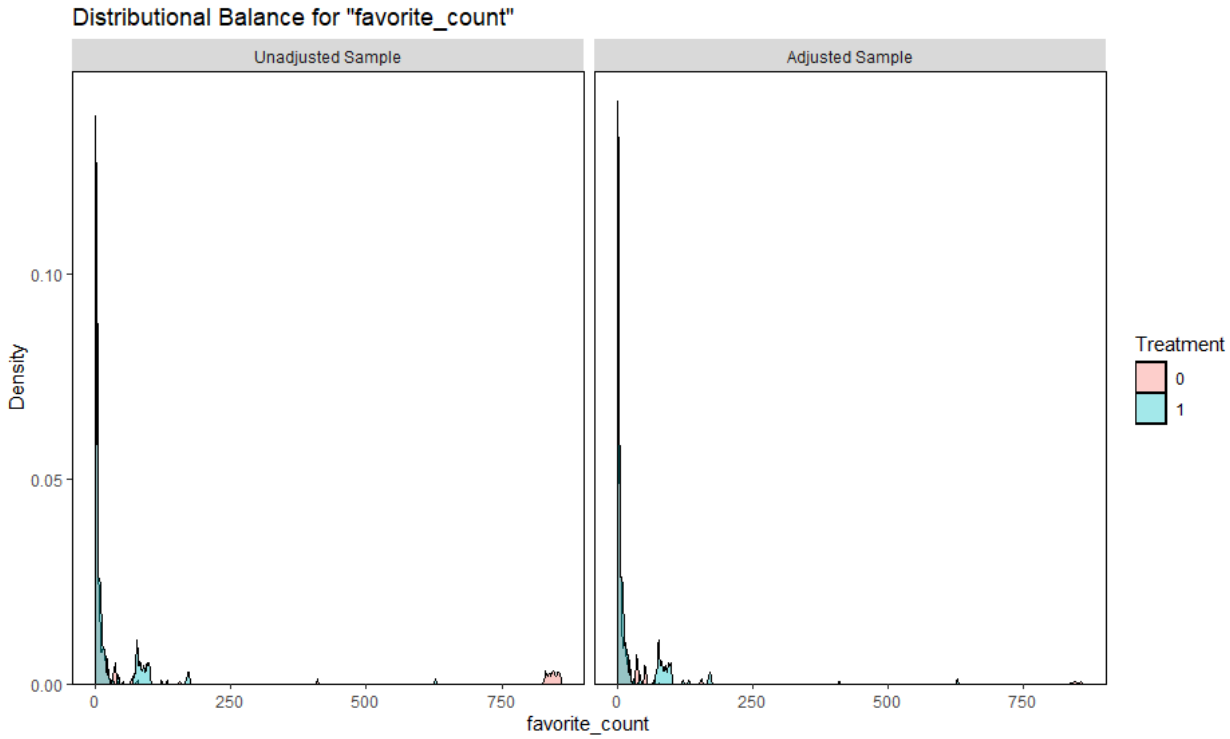


Figure A.9. (cont'd)

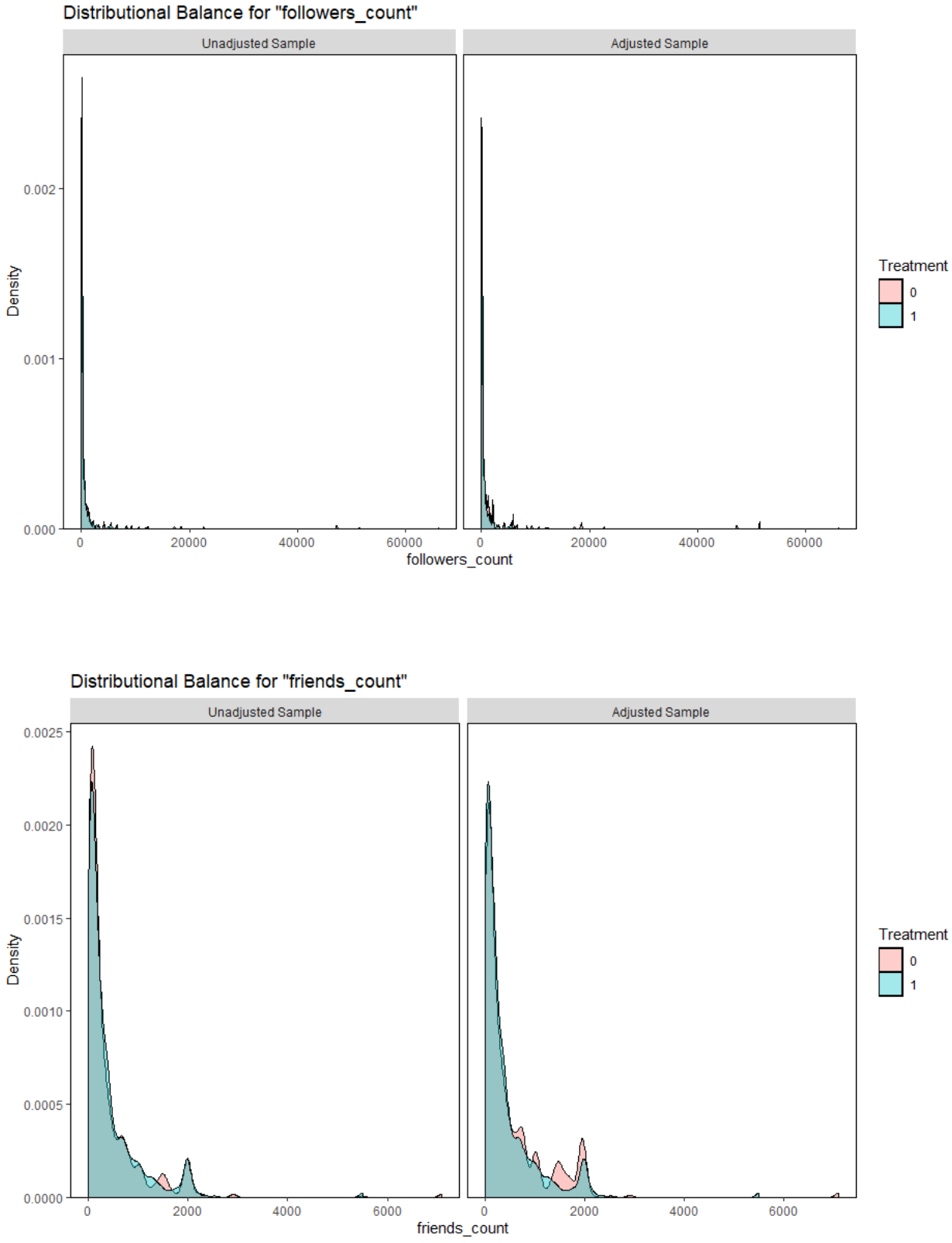


Figure A.9. (cont'd)

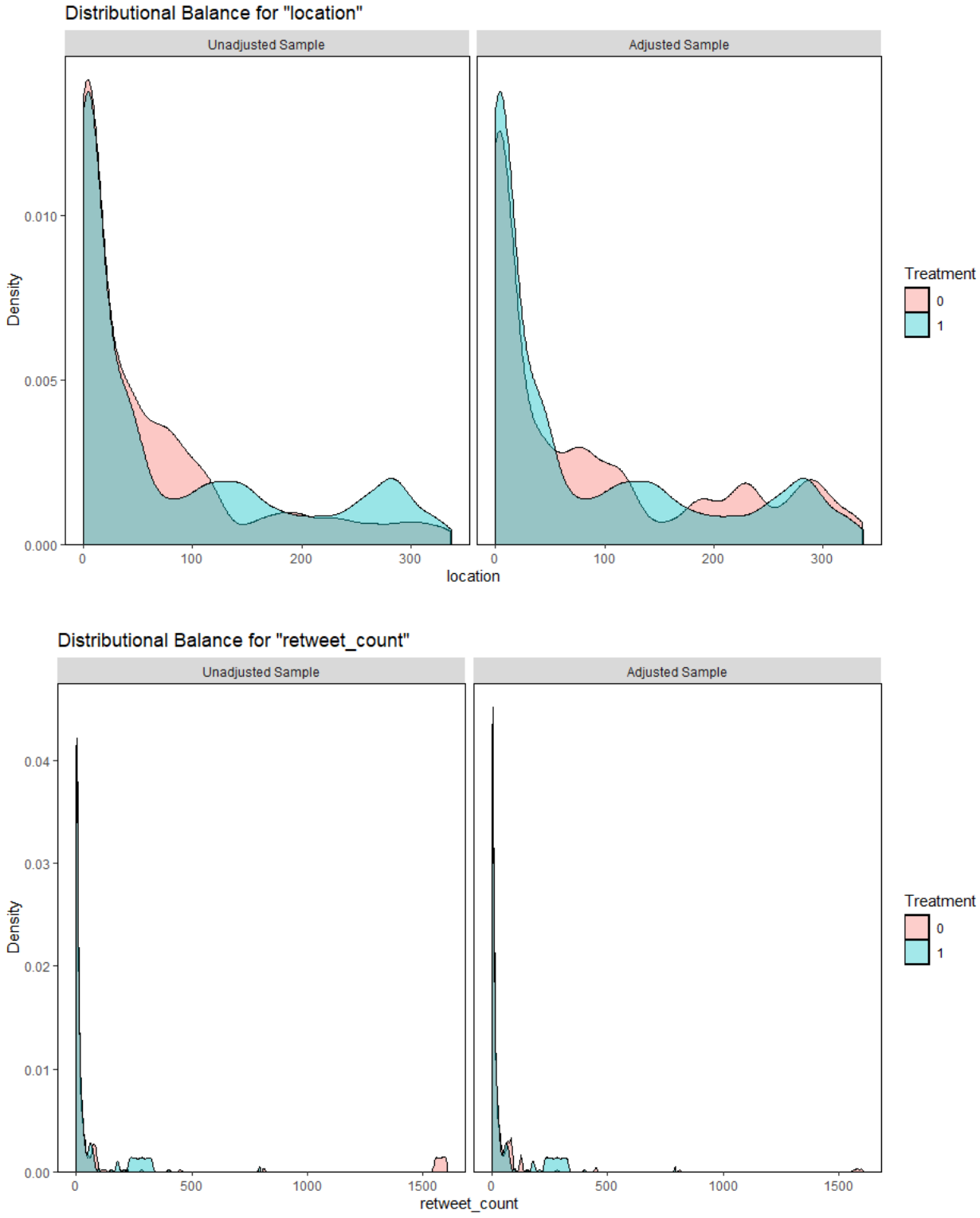


Figure A.9. (cont'd)

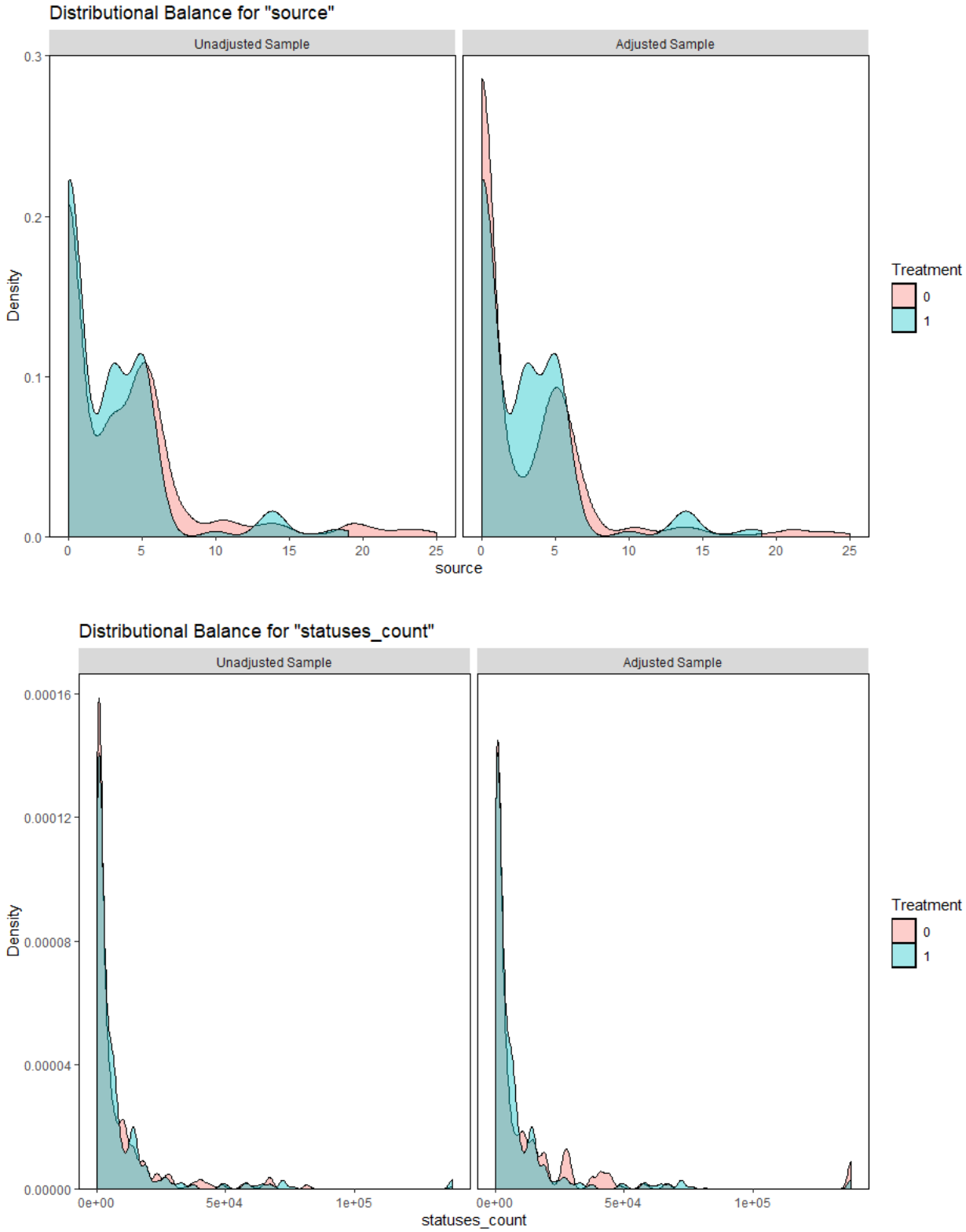
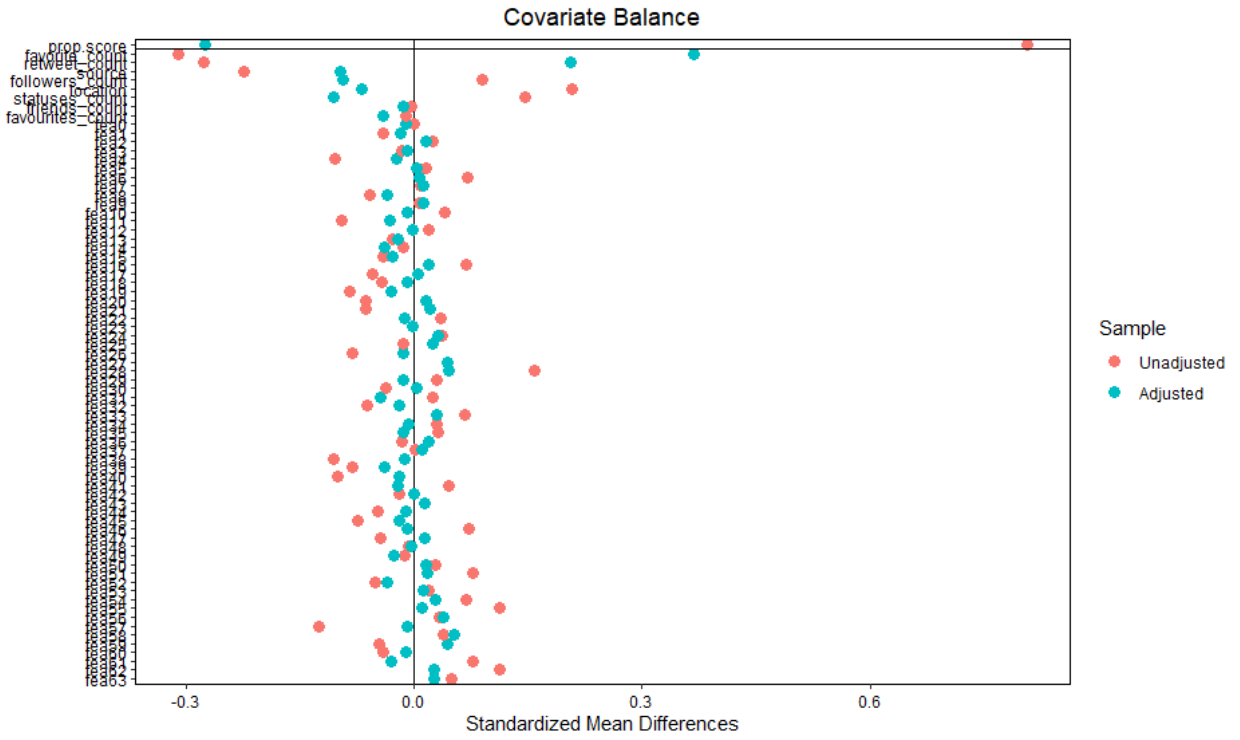


Figure A.10. Covariate Balanced Table for both theory based feature and latent features:



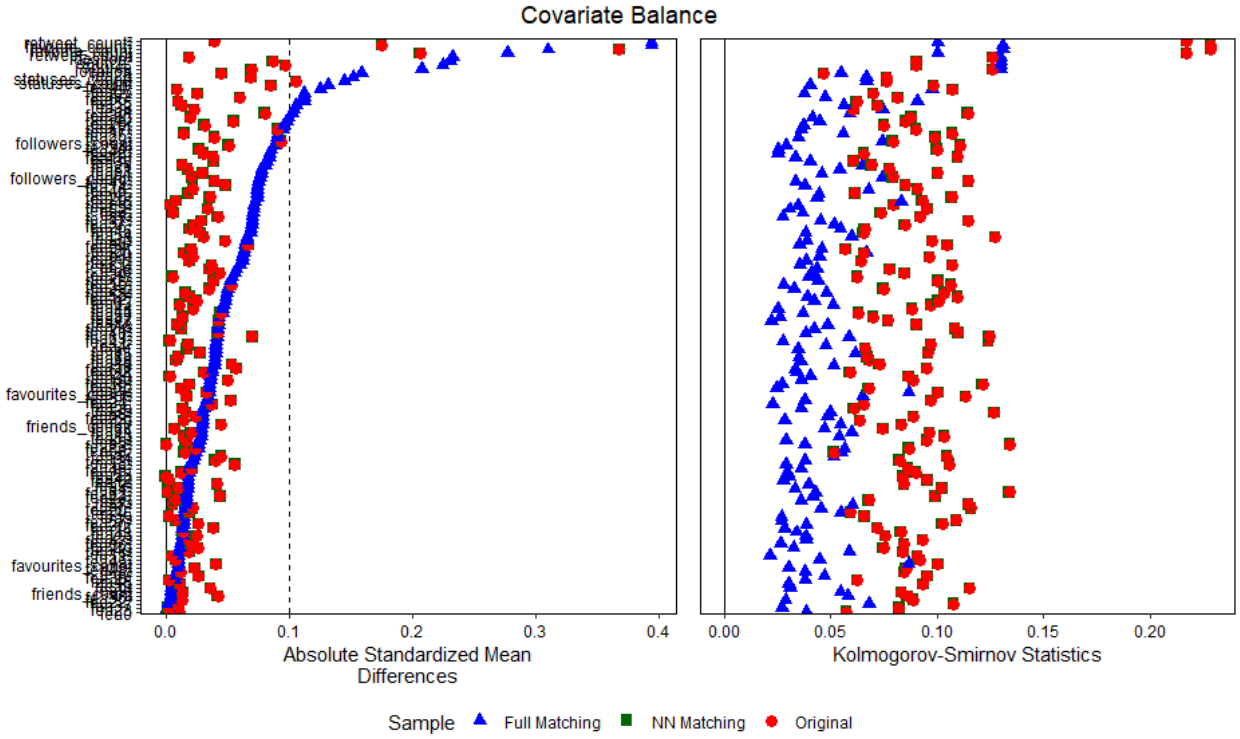


Figure A.11. Distribution Balance for each theory based feature

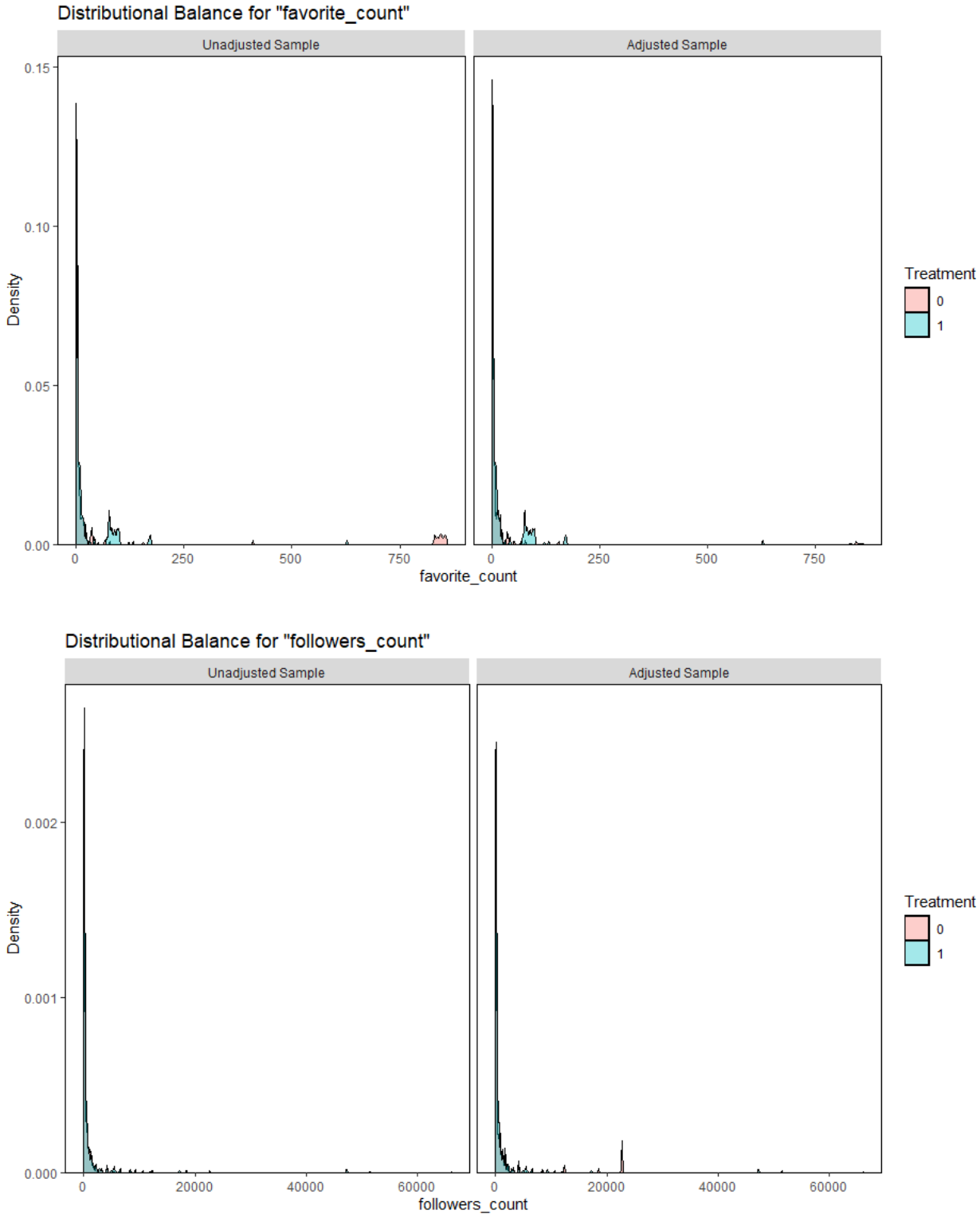


Figure A.11. (cont'd)

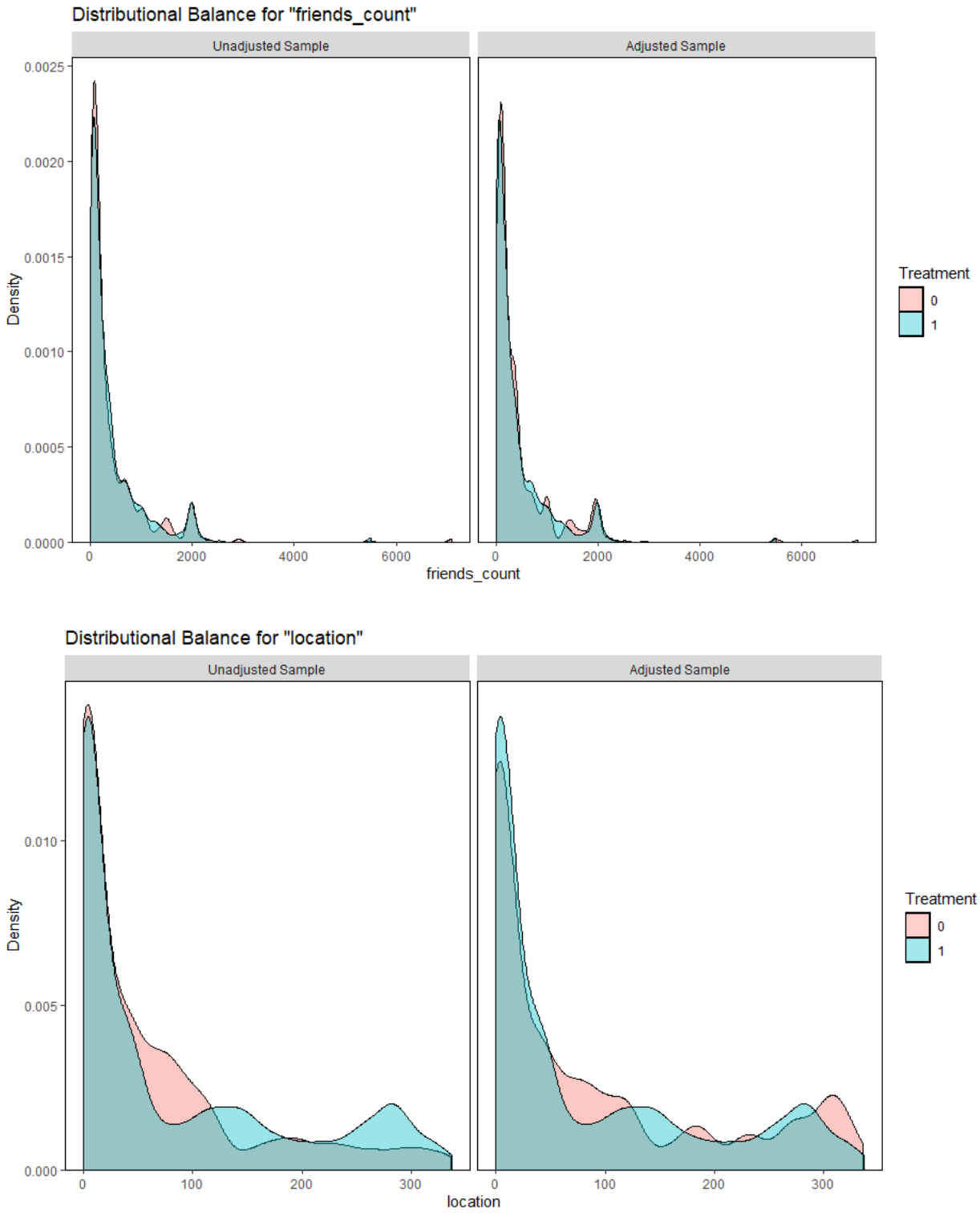


Figure A.11. (cont'd)

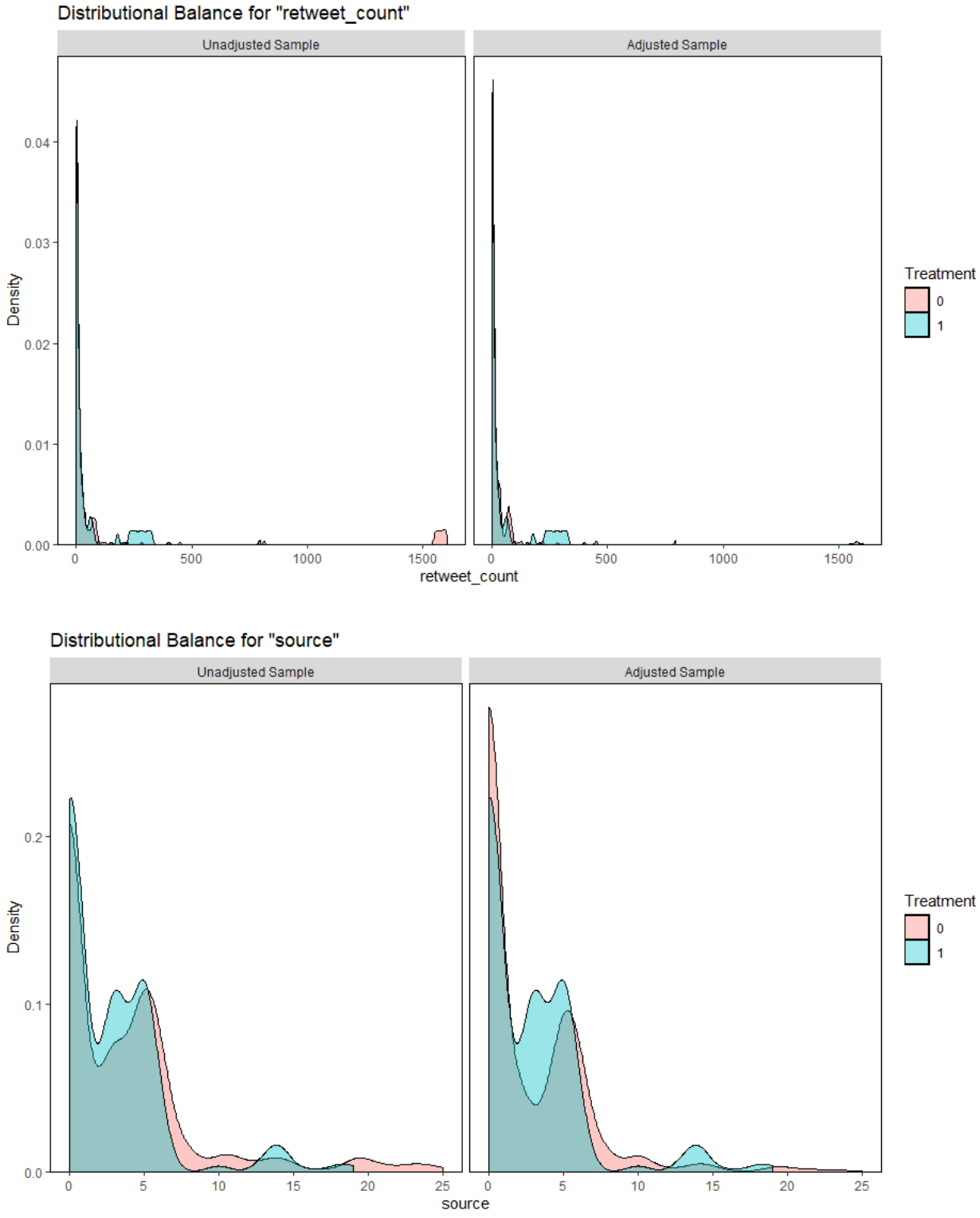


Figure A.11. (cont'd)

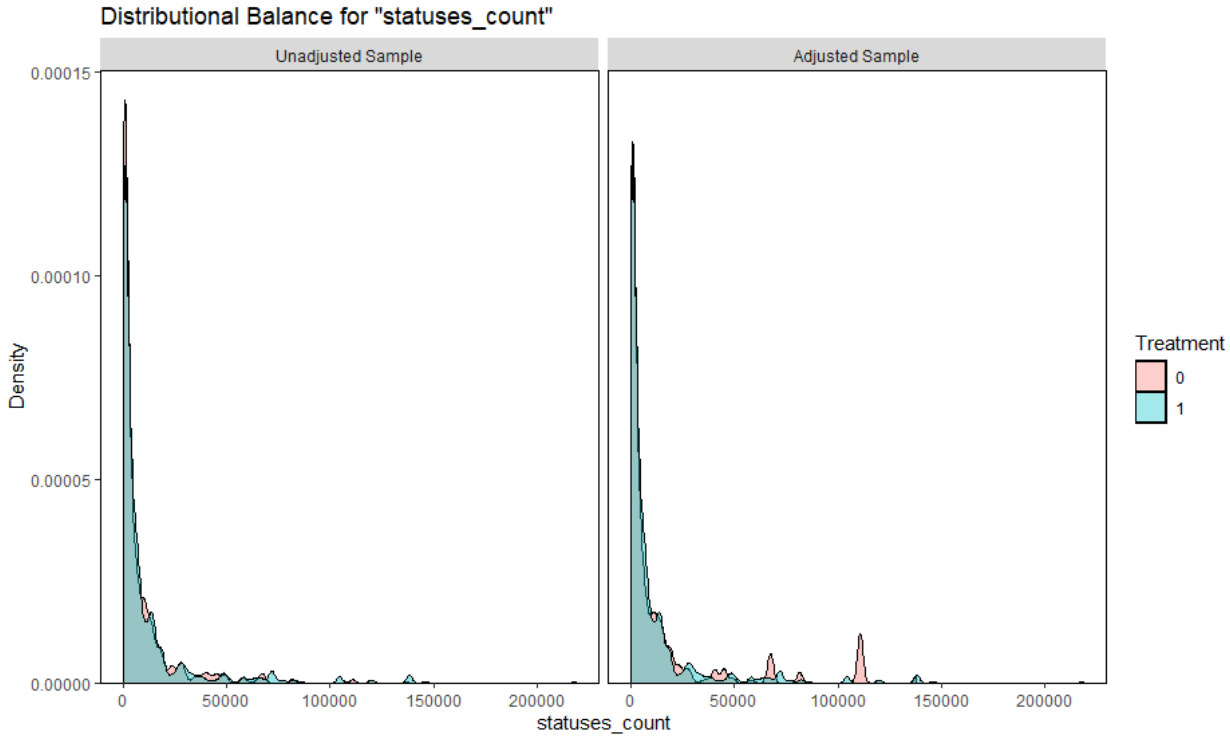


Figure A.12. Distribution Balance for each latent feature

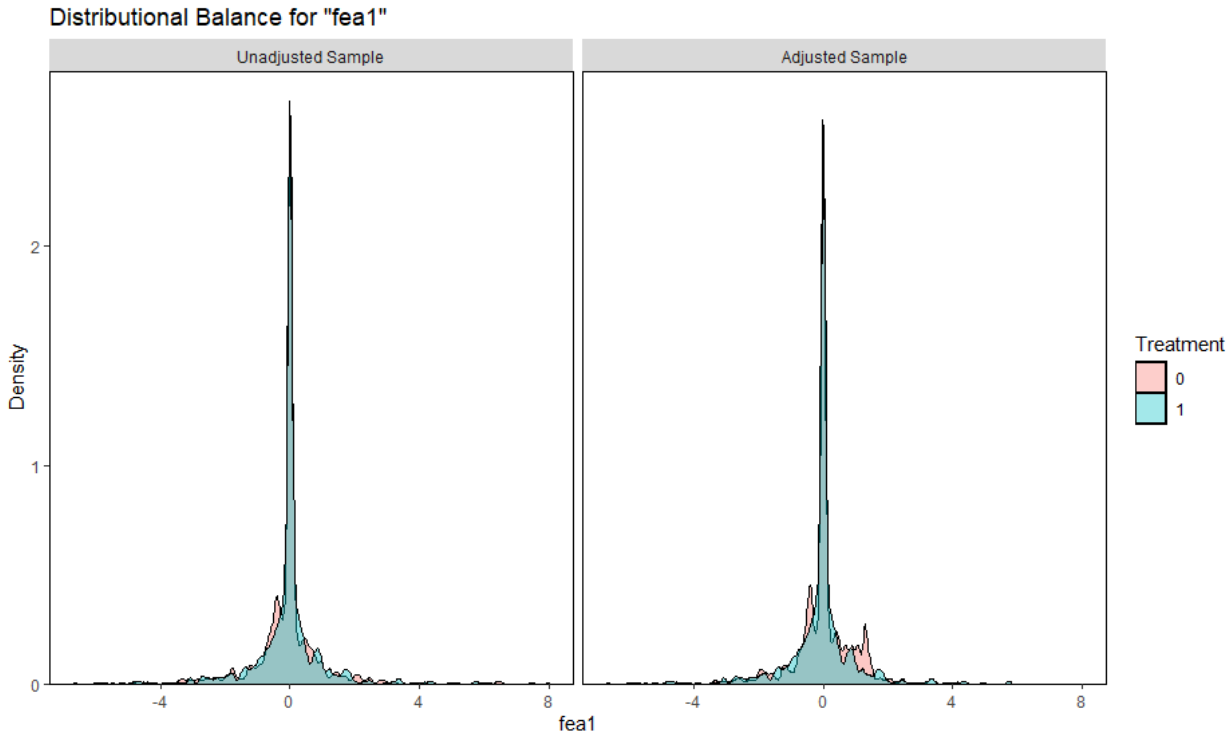


Figure A.12. (cont'd)

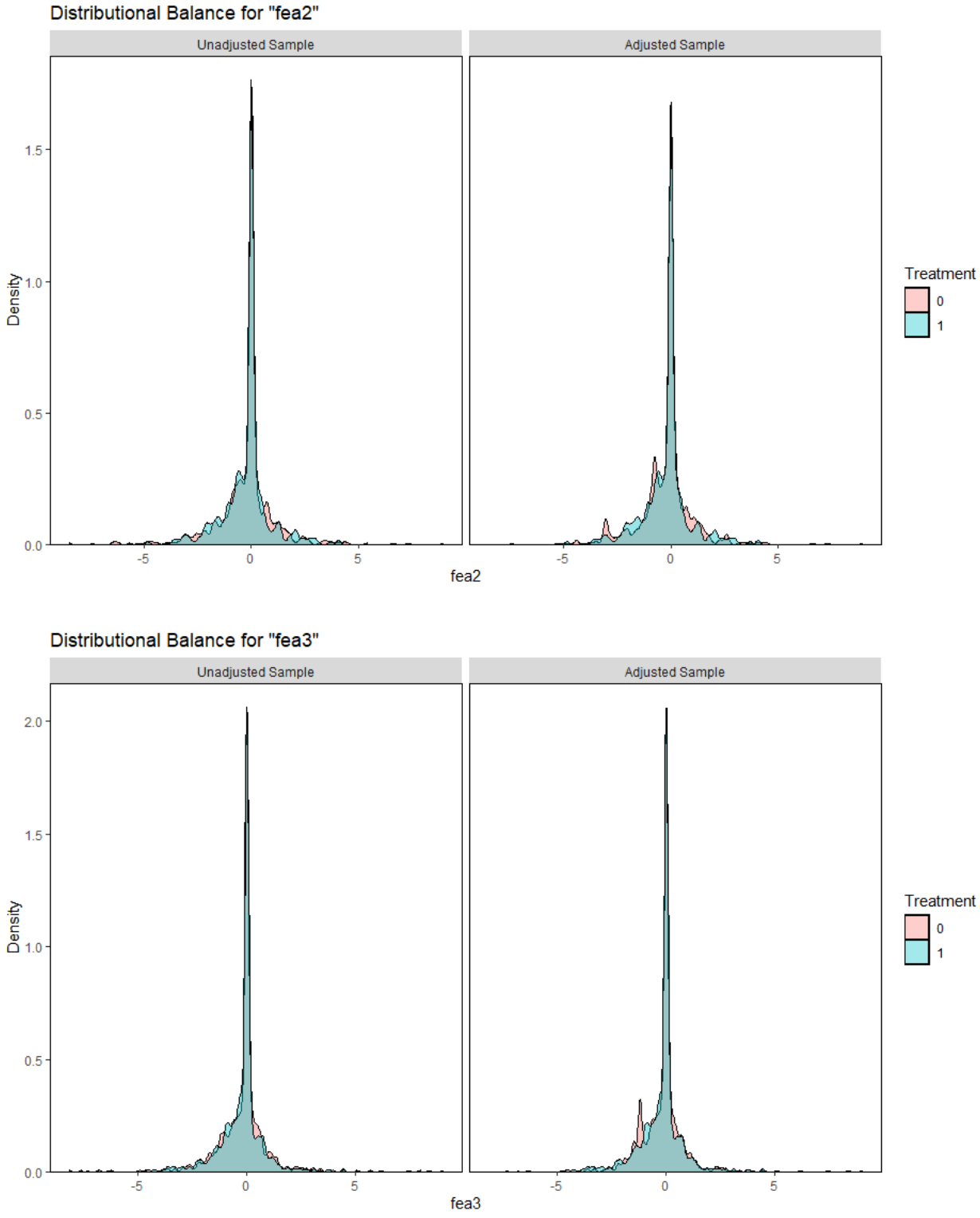


Figure A.12. (cont'd)

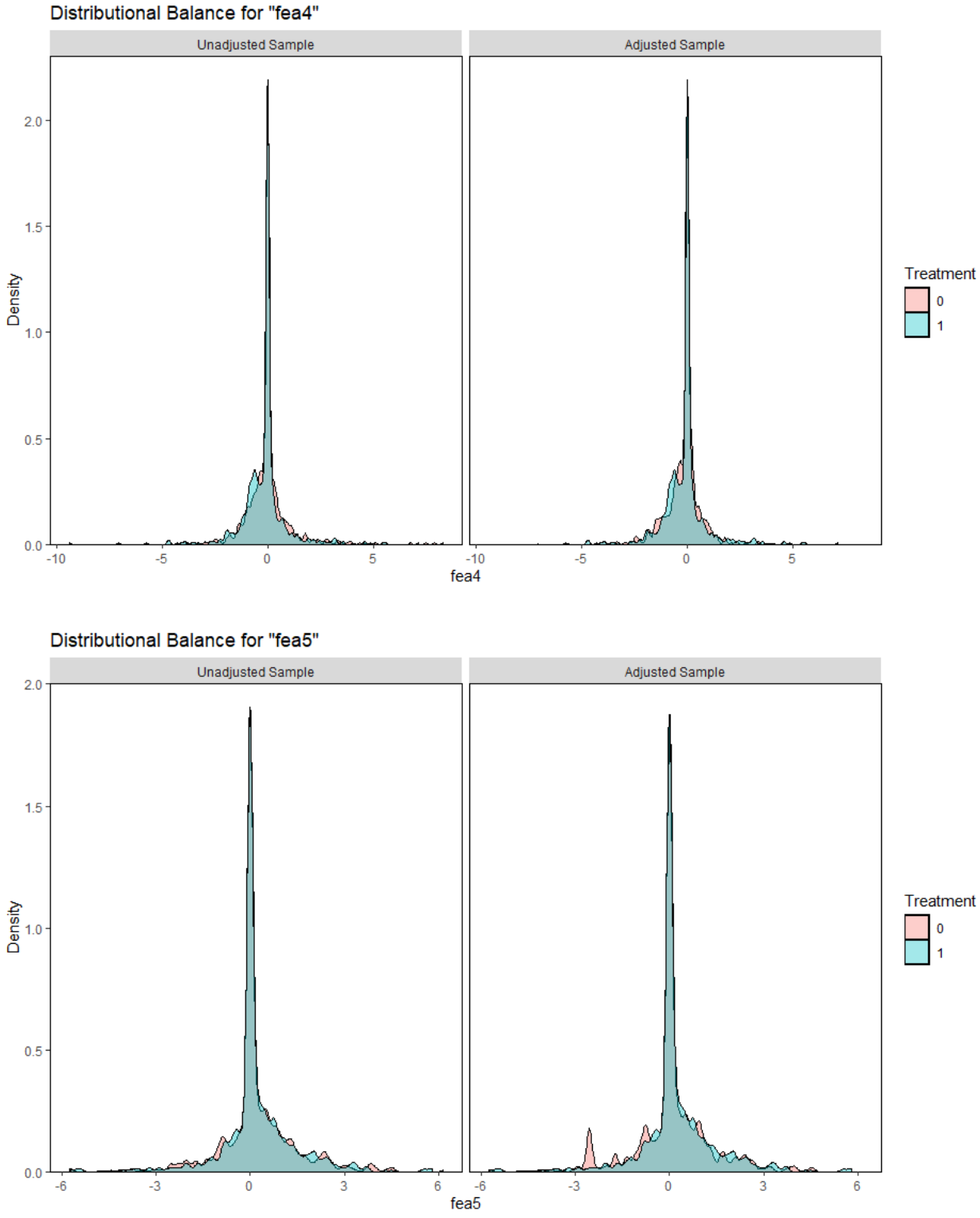


Figure A.12. (cont'd)

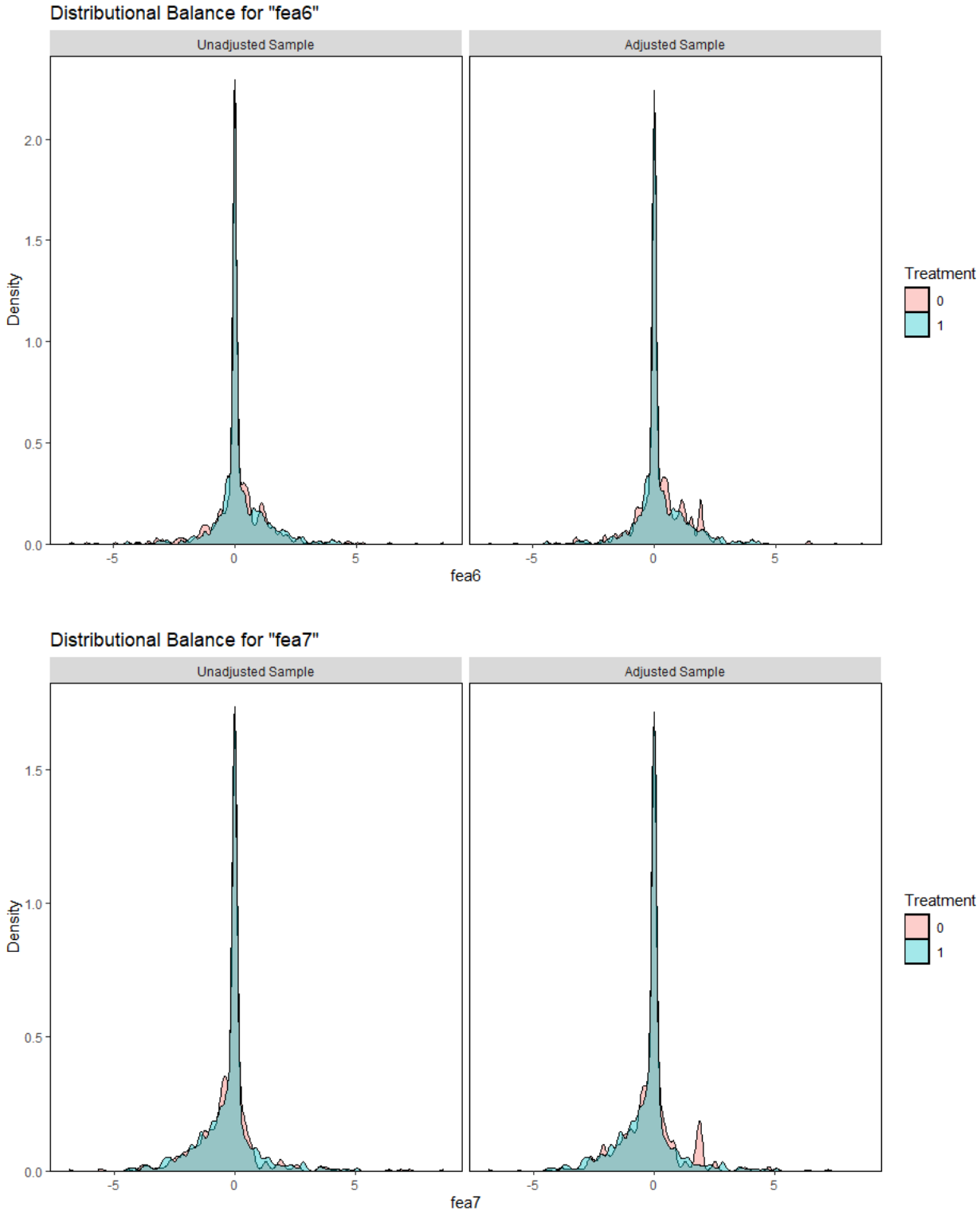


Figure A.12. (cont'd)

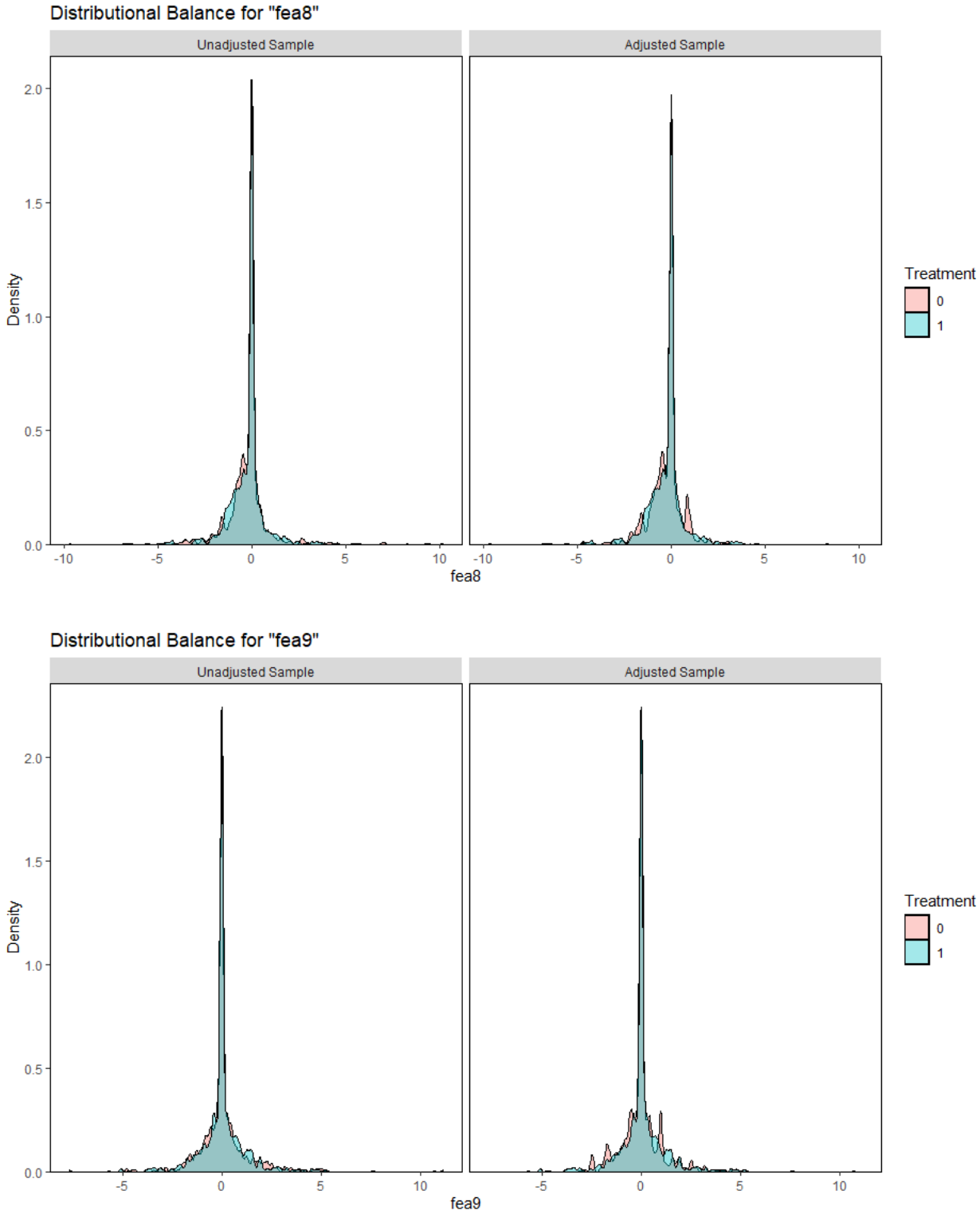


Figure A.12. (cont'd)

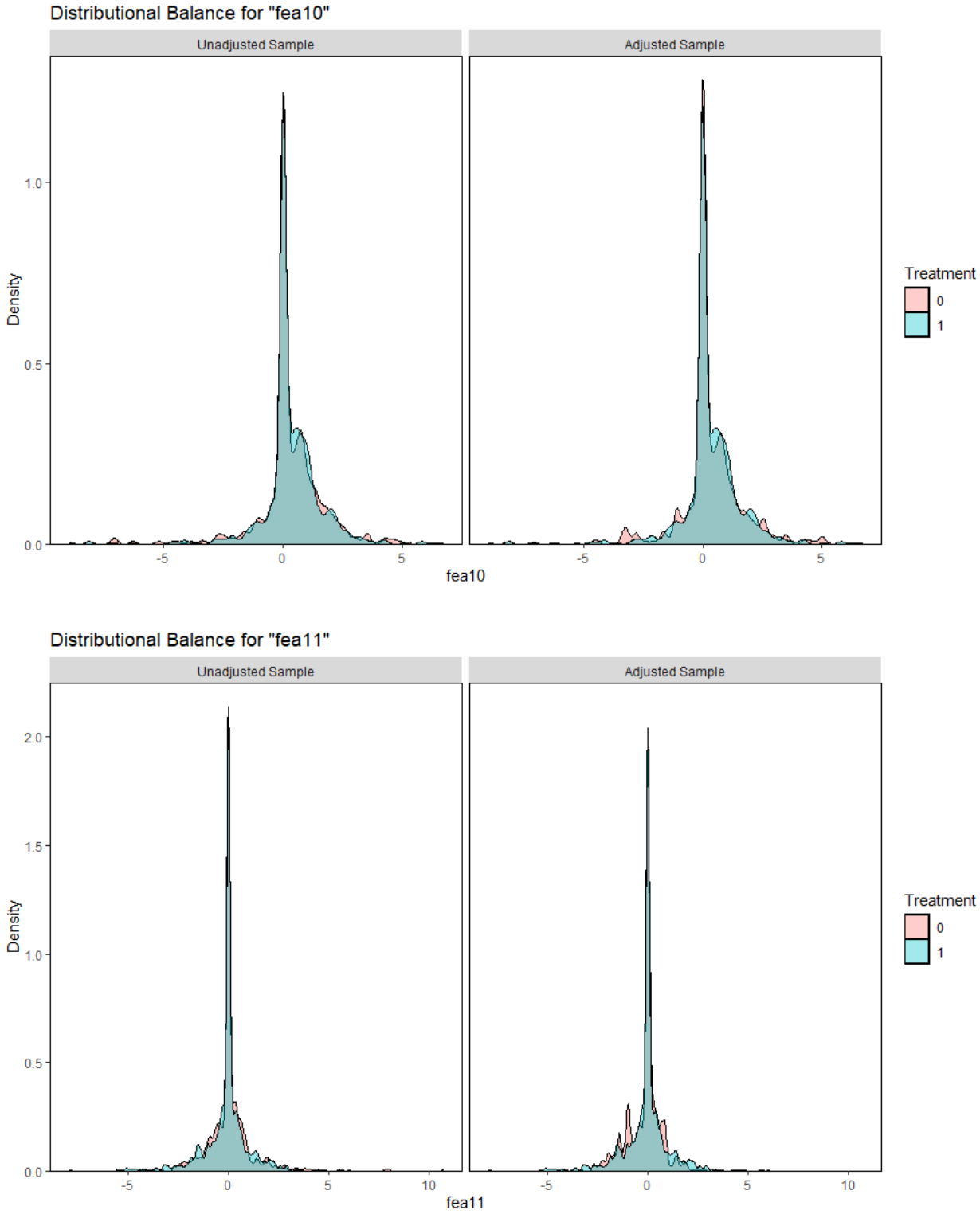


Figure A.12. (cont'd)

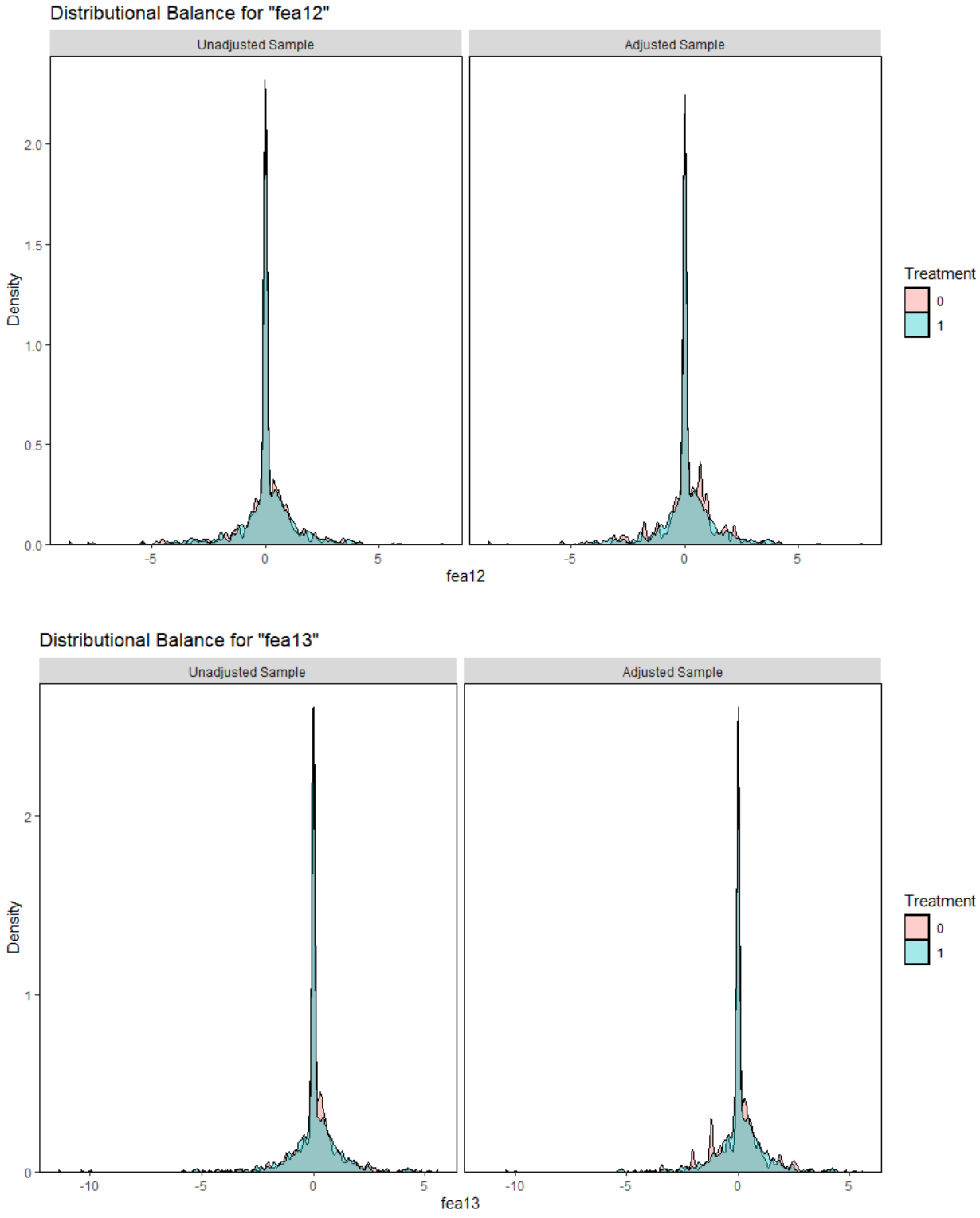
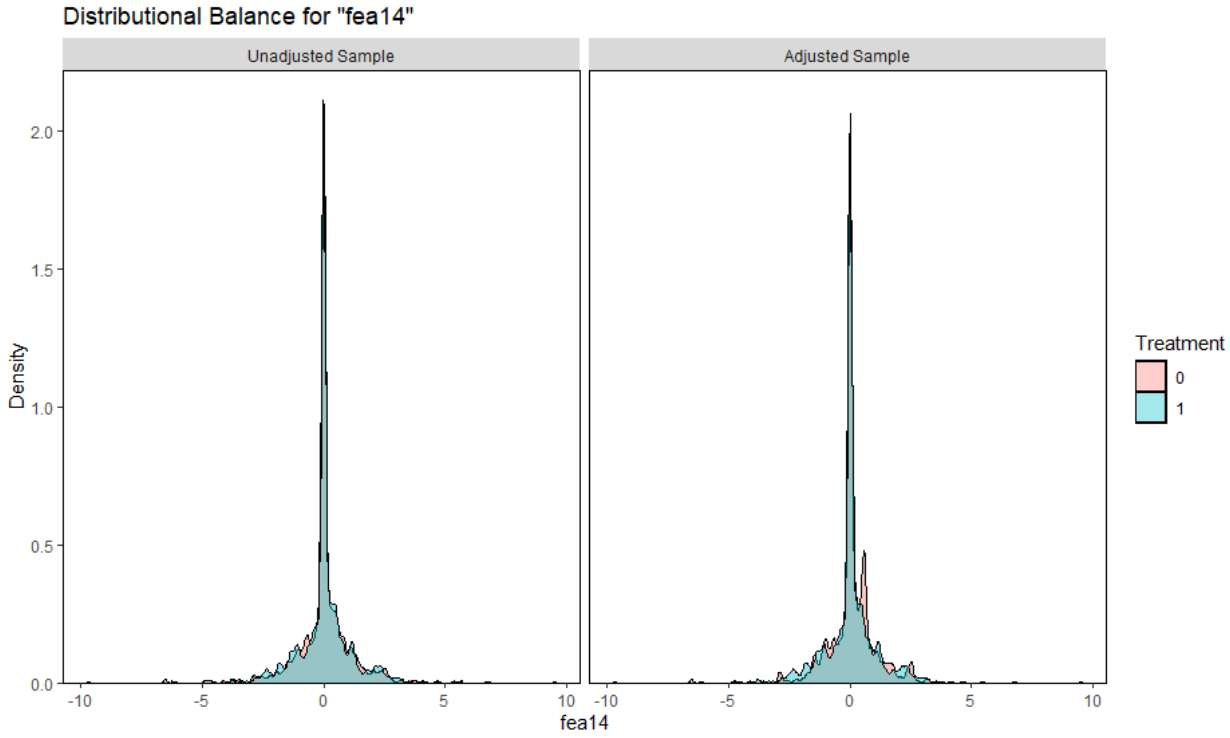


Figure A.12. (cont'd)



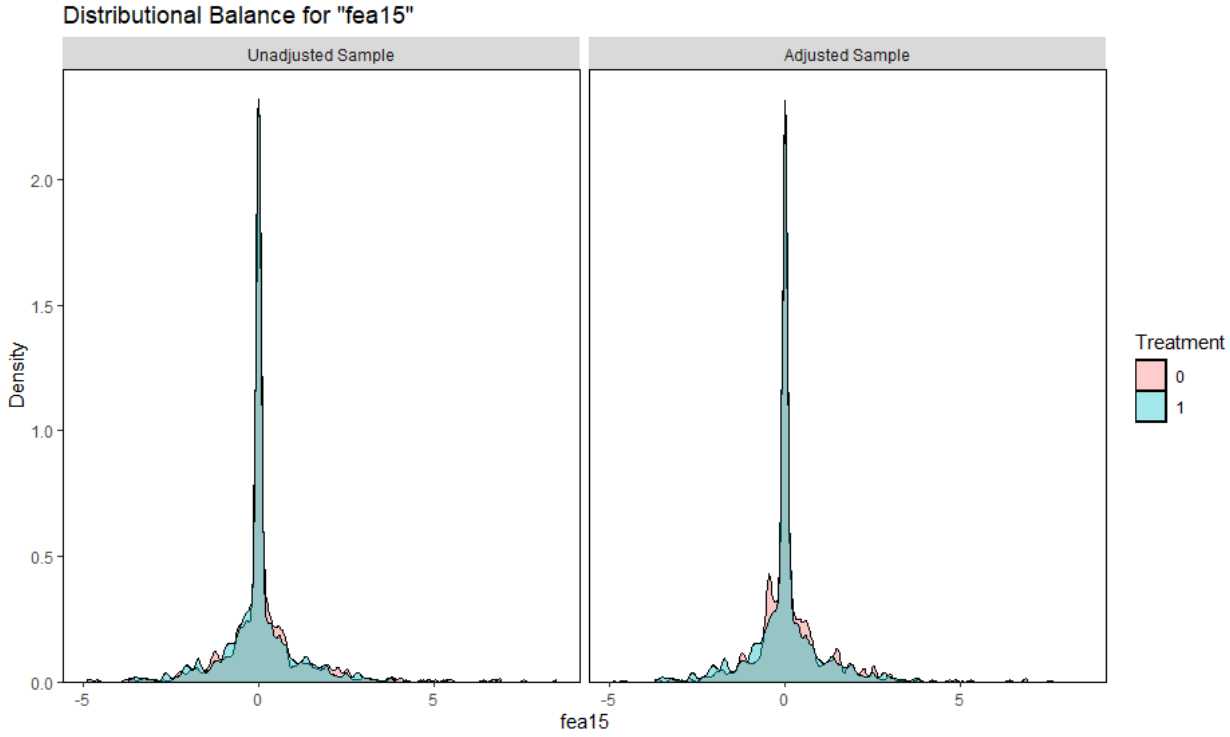
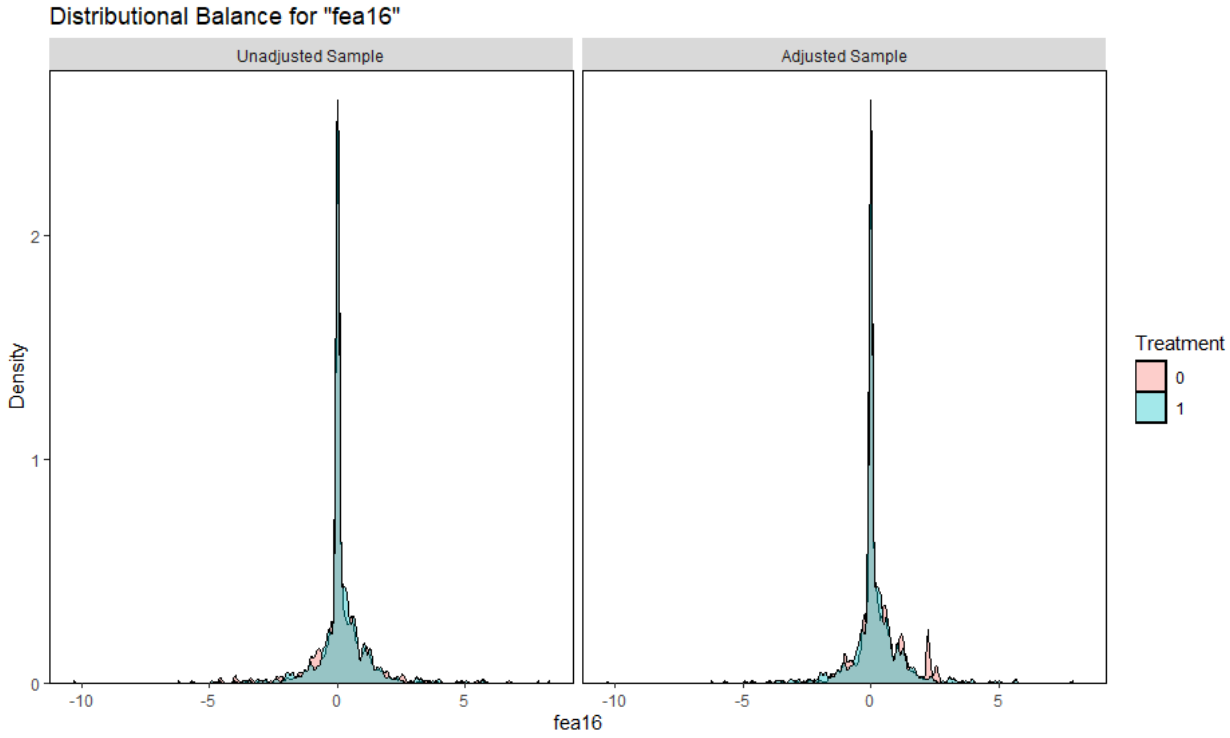


Figure A.12. (cont'd)



APPENDIX I. Notation in essay two

Table A.12. Notation in essay two

Notation	Explanation
u_μ, u_ν	Randomly choose user μ and ν
y	The ground truth of the sentiment scores
$\hat{y}$	The predicted truth of the sentiment scores
x_i^T	Original feature for node i with type t
$x_{sensitive}$	The sensitive attributes affects the fairness
Φ	Meta-path
h	Node feature
W_ϕ	Type-specific transformation
e_{ij}^ϕ	Importance of meta-path based on pair (i, j)
a_ϕ	Node level attention
α_{ij}^ϕ	Weight of meta path ϕ constructed on node pair (i, j)
$\mathcal{N}_\phi$	Neighbors based on meta path ϕ
S_ϕ	Semantic level node embedding
b	Semantic level attention
w_ϕ	Importance of meta-path ϕ
β_ϕ	Weight of meta path ϕ
Z	The final embedding

APPENDIX J. Data category for the election in social network

We follow the decoding strategy suggested from Bulsara and Singh (2017)

User level: politician, media, parties, religions (Yogi etc), Judge

Table A.13. User level node type

User Type	Example	Description
Politician	Modi, Gandhi, Jayalalithaa, etc	Politician account
Parties	BJP, AAP, INC etc	Parties account
Media	The Times of India, The Hindu, etc	Third Party Media account
Religions	Yogi, Swami, etc	Religion account

Function Level: Party, Activity

Table A.14. Function level node type

Function Type	Strategy focus	Description
Party	Past Success	Past achievement of the party
	Ideology	Ideology of the party
Activity	Slogan	The slogan party declare
	Issues	The issues party meets in current time
	Policies	The policy of party declare to make
	Promise	Promise to satisfy the voters in exchange of support

Table A.14. (cont'd)

	Fund	Funds to support the party
Candidate	Highlight Candidate	Highlight the advantage of Candidate
		Attack the competitor
	Public Relations	Internet Marketing
		Rallies and debates

Content level: tweet, topic, hashtag, policy. Please check the details in word embedding model part

Table A.15. Content level node type

Content Type	Example	Description
Tweet	#SadacharYatra The #BJP government had promised employment to 13 lakh in 2007 but only 80,000 got jobs. #SadacharYatra	The content of each tweet
Topic	job, government, lakh, get, bjp, political, provide, lose, claim, pass	The topic extract from each tweet
Hashtag	#SadacharYatra, #BJP	The hashtag included in each tweet
Policy	Employee Policy	The policy which the tweet declare

Table A.16. Data Coding Scheme

Tweet Type	Example	Description
Mention other user	@BJP	A tweet which is post or commented for mention some other users
Hashtag	#BJP	A hashtag which is used to connect to the popular trend
Retweet		A tweet which is used to repost from the prior tweet
Post tweet		The original tweet which is not used to mention, reply or retweet

APPENDIX K. Word Embedding Model

Table A.17. Corpus of the Tweet (first 20 rows)

['delhi', 'chief', 'minister', 'arvind', 'kejriwal', 'leave', 'residence', 'meet', 'breakingnow']
['nigerian', 'wild', 'animal', 'cancer', 'goa', 'bjp', 'minister', 'co', 'oxwlwf']
['congress', 'bjp', 'face', 'first', 'time', 'democracy', 'politic', 'aadmi_reminiscence', 'european_nationalism']
['angry', 'rahuls_statement', 'bjp', 'bald', 'bjp', 'man_lead', 'protest', 'bald', 'people', 'http', 'co', 'vqfgwqjwcc']
['narendra', 'modi', 'lead', 'bjp', 'government', 'moodys', 'co', 'sdxjiem', 'congress', 'call', 'moody', 'agent', 'sangh']
['interesting_point', 'sevanti_raise', 'aap', 'dharna', 'anarchy', 'demolition', 'babri_masjid', 'bjp', 'constitutional']
['bad', 'news', 'bjps', 'lok_sabha', 'ticket_hideously', 'corrupt', 'amp', 'allege', 'murder', 'fighting', 'ganga']
['aap', 'dharna', 'anarchy', 'demolition', 'babri_masjid', 'bjp', 'constitutional']
['feel_compelle', 'explain', 'position', 'aap', 'never', 'congress', 'bjp', 'party']
['think', 'tweet', 'antiaap', 'even', 'pro', 'bjp', 'call', 'tension', 'lol', 'traitor']
['bad', 'news', 'bjps', 'lok_sabha', 'ticket_hideously', 'corrupt', 'amp', 'allege', 'murder', 'fighting', 'ganga']
['donate', 'today', 'co', 'ssfqetb', 'modi', 'pm', 'co', 'eofpnub']
['angry', 'rahuls_statement', 'bjp', 'bald', 'bjp', 'man_lead', 'protest', 'bald', 'people', 'http', 'co', 'vqfgwqjwcc']
['membership_drive', 'bjp', 'shud', 'limited', 'election', 'purpose', 'shud', 'employ', 'right', 'process', 'change', 'system']
['delhi', 'lucky', 'arvind', 'kejriwal', 'cm_hitte', 'road', 'accountable_police']
['interesting_point', 'sevanti_raise', 'aap', 'dharna', 'anarchy', 'demolition', 'babri_masjid', 'bjp', 'constitutional']
['sir', 'importantly', 'ppl', 'want', 'know', 'bjp', 'quiet', 'allegation', 'shinde', 'support', 'dawood', 'friend', 'hafta', 'collection']
['bjp', 'pm', 'candidate', 'narendra', 'modi', 'set', 'address', 'mega', 'rally', 'gorakhpur', 'today', 'live', 'coverage', 'air', 'country']
['share', 'pic', 'independent', 'mp', 'bihar', 'join', 'bjp', 'support', 'bjp', 'bihar', 'increase', 'consistently', 'co', 'hjvveilz']
['dear_chhidu', 'do', 'know', 'muslim', 'rehman', 'amp', 'bjp', 'muslim', 'bjp']

Table A.18. The most frequent Bi-gram features of different Parties (Top 20)

AAP	BJP	INC
('us', 'several')	('support', 'BJP')	('several', 'scams')
('Babri', 'Masjid')	('BJP', 'led')	('the', 'state')
('the', 'AAP')	('about', 'stabilitdevelopment')	('vote', 'share')
('Amethi', 'today')	('GDP', 'growth')	('Rajiv', 'Gandhi')
('AAP', 'Dharna')	('Narendra', 'Modis')	('the', 'demolition')
('Interesting', 'point')	('for', 'BJP')	('techie', 'behind')
('NDA', 'brought')	('BJP', 'government')	('ppl', 'manage')
('Nation', 'opinion')	('Uttar', 'Pradesh')	('MOTNPoll', 'Vote')
('women', 'members')	('Chandra', 'Bose')	('an', 'Annarchist')
('speaking', 'to')	('India', 'Today')	('Congress', 'VP')
('Perhaps', 'biggest')	('NDA', 'GDP')	('Gorakhpur', 'rally')
('2012', 'reelection')	('Sabha', 'projection')	('governments', 'decision')
('but', 'improving')	('BJP', 'surge')	('own', 'corruption')
('JoinAAP', 'SaveNation')	('Opinion', 'Poll')	('Doctor', 'said')
('small-scale', 'industries')	('share', 'in')	('waste', 'vote')
('Congress', 'does')	('to', 'Gujrat')	('formed', 'alliance')
('elitist', 'swipe')	('.Congress', 'debacle')	('handedly', 'revived')
('BJPs', 'anarchy')	('Elections', 'carries')	('News-Nielsen', 'Opinion')
('Today', 'poll')	('economic', 'growth')	('against', 'poverty')
('against', 'Vadra')	('BJP-MDMK', 'alliance')	('challenged', 'youths')

Table A.19. The most frequent Tri-gram features of different Parties (Top 20)

AAP	BJP	INC
('Dharna', 'is', 'anarchy')	('I', 'support', 'BJP')	('When', 'Manmohan', 'Singh')
('Interesting', 'point', 'Sevanti')	('of', 'my', 'country')	('Govt', 'of', 'India')
('Self', 'Help', 'Groups')	('GDP', 'growth', 'rate')	('Mahila', 'Vikas', 'Pariyojana')
('LIVE', 'will', 'be')	('BJP', 'in', 'UP')	('Funds', 'for', 'Indira')
('Gorakhpur', '#', 'NaMoInGKP')	('The', 'techie', 'behind')	('Vijay', 'Sankalp', 'Rally')
('Now', 'we', 'know')	('NDA', 'GDP', 'growth')	('Congress', 'Vice', 'President')
('News-Nielsen', 'Opinion', 'Poll')	('brought', 'more', 'development')	('District', 'Level', 'Meet')
('Nation', 'opinion', 'poll')	('for', 'Narendra', 'Modis')	('Vice', 'President', ' Rahul')
('MOTNPoll', 'Lok', 'Sabha')	('Narendra', 'Modi', 'pays')	('Vote', 'share', 'of')
('the', 'total', 'promised')	('party', 'can', 'increase')	('India', 'Today', 'Mood')
('for', 'Ahmedabad', 'Civil')	('sweep', 'the', 'state')	('praises', 'the', 'lea')
('UP', 'poll', 'tracker')	('Modi', 'pays', 'tribute')	('answer', 'on', 'misuse')
('Ravi', 'Shankar', 'Prasad')	('total', 'promised', 'jobs')	('crore', 'central', 'fund')
('Ahmedabad', 'Civil', 'Hospita')	('twitter', 'accusing', 'us')	('DusDusKiDaud', 'Srinagar', 'NDA')
('empowerment', 'of', 'women')	('electiontracker', 'shows', 'BJP')	('When', 'Arvind', 'Kejriwal')
('Chandra', 'Bose', 'on')	('on', 'the', 'upswing')	('today', 'at', 'Dandi')
('origins', 'have', 'embarrassed')	('SadacharYatra', 'against', 'Guj')	('Gandhi', 'addressing', 'women')
('Vijay', 'Shankhnad', 'Rally')	('to', 'sell', 'land')	('jobs', 'created', 'in')
('be', 'an', 'Annarchist')	('people', 'during', 'his')	('Priyanka', 'Gandhi', 'speaking')
('in', 'Tamil', 'Nadu')	('groups', 'and', 'representations')	('bad', 'in', 'law')

Table A.20. Top 10 keywords for 5 Topics of different Parties

Topic No.	AAP	BJP	INC
1	0.626 "do" 0.079 "share" 0.038 "know" 0.031 "road" 0.000 "create" 0.000 "nda" 0.000 "rule" 0.000 "yashwant_sinha" 0.000 "price_rise" 0.000 "come"	0.584 "bjp" 0.119 "india" 0.113 "job" 0.065 "talk" 0.054 "time" 0.032 "leader" 0.005 "new" 0.004 "alliance" 0.000 "nda" 0.000 "growth"	0.554 "medium" 0.077 "cong" 0.073 "show" 0.071 "pay" 0.065 "work" 0.026 "ppl" 0.005 "interest" 0.000 "trust" 0.000 "house" 0.000 "doubt"
2	0.110 "candidate" 0.105 "live" 0.096 "set" 0.091 "rally" 0.041 "hope" 0.038 "dharna" 0.022 "gorakhpur" 0.021 "anarchy" 0.019 "address" 0.014 "babri_masjid"	0.451 "kejriwal" 0.263 "arvind" 0.000 "high" 0.000 "reason" ' 0.000 "ka" 0.000 "suroor" 0.000 "start" 0.000 "career" 0.000"aam_aadmi" 0.000"communal	0.233 "sonia" 0.209 "call" 0.186 "explain" 0.077 "think" 0.072" even" 0.046 "get" 0.026 "voter" 0.016 "tweet" 0.012" namoingkp" 0.011 "pro

Table A.20. (cont'd)

3	0.236 "choice" 0.006 "sack" 0.006 "later" 0.005 "evidence_mount" 0.005"miscalculation_pressure" 0.000 "nda" 0.000 "rule" 0.000 "make" 0.000 "next" 0.000 "vote"	0.485 "singh" 0.349 "manmohan" 0.022 "mp" 0.000 "prime" 0.000 "dr" ' 0.000 "replug" 0.000 "prove" 0.000 "indeed" 0.000 "tev_evrx" ' 0.000 "hire"	0.382 "party" 0.262 "lakh" 0.122 "day" 0.063 "join" 0.000 "create" 0.000 "nda" 0.000 "yashwant_sinha" 0.000 "rule" 0.000 "price_rise" 0.000 "year
4	0.386"aap" 0.198"want" 0.179"money" 0.031"sir" 0.000"card" ' 0.000"bill" 0.000"communal_violence" 0.000"expose" 0.000"appeasement" 0.000"tell	0.472"upa" 0.082"put" 0.077"pm" 0.074"govt" 0.046"ask" 0.044"india" 0.035"lead" 0.034"support" 0.033"country" 0.025"today"	0.721"amp" 0.066"need" 0.064"much" 0.028"thing" 0.001"proud" 0.000"economy" 0.000"ground" 0.000"approach" 0.000"run" 0.000"wrong"

Table A.20. **(cont'd)**

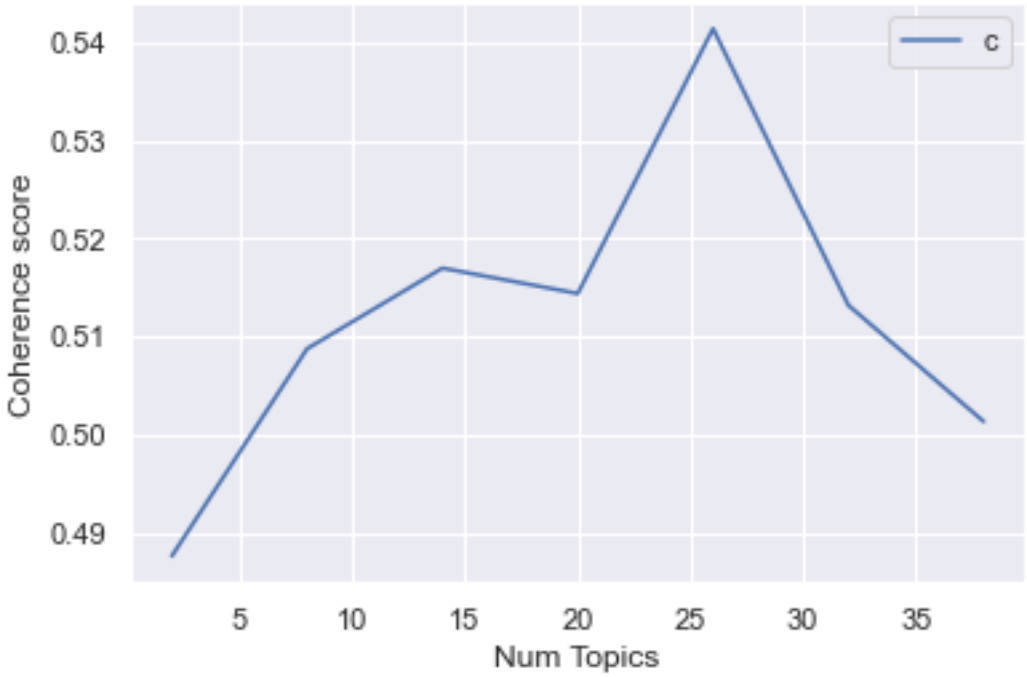
5	0.321"congress"	0.211 "bjps"	0.394 "meet"
	0.235"say"	0.151 "first"	0.295 "delhi"
	0.144"gujarat"	0.141 "politic"	0.118 "minister"
	0.099"riot"	0.086 "face"	0.035 "chief"
	0.092"muslim"	0.076 "news"	0.000 "university"
	0.037"s"	0.074 "corrupt"	0.000 "tirupati"
	0.001"maharashtra"	0.064 "bad"	0.000"international_islamic"
	0.000"vote"	0.037 "lok_sabha"	0.000 "announce"
	0.000"ydwtppfyo"	0.007 "allege"	0.000 "seriously_oppose"
	0.000"unite	0.001 "fighting"	0.000 "setup"

The table shows the top 10 keywords that contribute to each topic. The weights reflect how important a keyword is to that topic.

Table A.21. Topic Perplexity and Coherence Score

Measurement	Descriptions	Result
Perplexity	A measure of how good the model is. The lower the better	-18.065618014400595
Coherence Score	Measures score a single topic by measuring the degree of semantic similarity between high scoring words in the topic	0.3486032341447605

Figure A.13. Different Topic Numbers vs Coherence Scores on each topic model



The figure shows that when topic number is 26, we get the highest coherence scores

Table A.22. Finding the dominant topics in each sentence

Document_No	Dominant_Topic	Topic_Perc_ Contrib	Keywords	Text
0	13.0	0.458999991 41693115	kejriwal, arvind, first, mr, take, co, minister, law, delhi, word	Delhi Chief Minister Arvind Kejriwal leaves his residence to meet Lieutenant Governor Najeeb Jung #BreakingNow #AAPInstantJustice
1	14.0	0.387199997 9019165	bjp, economy, alliance, come, co, cute, break, policy, keep, performance	Nigerians are like wild animals and cancer - Goa BJP Minister http://t.co/oxWLWF11V6
2	20.0	0.401399999 85694885	modi, co, narendra, congress, deliver, http, face, politic, friend, never	Congress,BJP are facing first time the Democracy,Politics of the Extraordinary Aam Aadmi.Reminiscence of European Nationalism!!!
3	2.0	0.583500027 6565552	https, corruption, co, other, bjp, amp, ever, proof, address, freedom	RT Angry over Rahuls statement that BJP can sell combs to bald, BJP man leads protest of bald people. http://t.co/vQfgwqjWcc
4	14.0	0.597400009 6321106	bjp, economy, alliance, come, co, cute, break, policy, keep, performance	Narendra #Modi-led BJP government can lift mood: Moodys http://t.co/9SdXjIeM2X Now #Congress will call Moody an agent of the Sangh !
5	11.0	0.903100013 7329102	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	Interesting point Sevanti raises. If the AAP Dharna is anarchy, then what was the demolition of the Babri Masjid by the BJP? Constitutional?
6	27.0	0.763800024 9862671	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	Bad news if BJPs Nishank getting Lok Sabha ticket.Hideously corrupt & alleged to be behind the murder of Swami Nityanand fighting for Ganga
9	11.0	0.878899991 5122986	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	If the AAP Dharna is anarchy, then what was the demolition of the Babri Masjid by the BJP? Constitutional? #JoinAAPSaveNation
10	12.0	0.599900007 2479248	explain, instead, mp, speech, expose, stop, thing, congress, issue, bjp	Why does feel compelled to explain its position on #AAP? Since it never did for #Congress, #BJP or any other party?
11	19.0	0.334899991 75071716	aap, delhi, bjp, co, congress, thank, true, act, non, rti	I think 90% of her tweets are antiAAP . Not even 1% pro BJP . This is called Tension Lol . Traitor

Table A.22. (cont'd)

12	27.0	0.8924000263214111	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	Donate Today for Better Tomorrow! Donate online at http://t.co/4SsfQetb5b #Modi4PM http://t.co/y44EoFpnub
13	4.0	0.8446999788284302	seat, bjp, put, trust, share, win, tv, vote, election, pay	These membership drives by BJP shud not be limited 2 just for election purposes. This shud be employed right thru the process. Change system
14	13.0	0.8787999749183655	kejriwal, arvind, first, mr, take, co, minister, law, delhi, word	Delhi is Lucky to Have Arvind Kejriwal ; The CM hitting the roads for accountable Police .#JoinAAPSaveNation
15	2.0	0.5997999906539917	https, corruption, co, other, bjp, amp, ever, proof, address, freedom	AAP पार्टी लोकसभा चुनाव जितने के लिए नहीं BJP को हराने और UPA को जिताने के लिए लड़ रही है #QuitAAPsaveNation
16	20.0	0.1899999976158142	modi, co, narendra, congress, deliver, http, face, politic, friend, never	Sir more importantly ppl want to know Y BJP is quiet on allegations that Shinde supports Dawood friends? Hafta collection
17	27.0	0.8440999984741211	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	BJP's PM candidate Narendra Modi is all set to address a mega rally in UP's Gorakhpur today, live coverage to be aired in 37 countries
18	15.0	0.6037999987602234	india, co, bjp, namo, fund, wrong, listen, avg, today, source	Sharing pic of two independent MPs from Bihar joining BJP. Support for BJP in Bihar is increasing consistently. http://t.co/hJVVeIlZ8h
19	31.0	0.6873000264167786	bear, gujarat, bjp, s, muslim, riot, narendramodi, congress, report, modi	Dear #Chhidu dont you know that ??? The only muslims who won in Rajasthan- Habibur Rehman & Yunus Khan, both from BJP. #Muslims are with BJP
20	31.0	0.4487000107765198	bear, gujarat, bjp, s, muslim, riot, narendramodi, congress, report, modi	BJP and Cong are not realizing that more they throw mud at more will fall on them.Ppl will realize they hv no substance.

Table A.23. Most Representative Topics for Each Document

Topic_Num	Topic_Perc_Contrib	Keywords	Text
0.0	0.8788999915122986	gandhi, rahul, co, meet, sonia, amethi, congress, http, look, visit	#Rahul #Gandhi meeting Ex-servicemen personnel in Amethi today. http://t.co/CopS6rb4uL http://t.co/fZatfpwJJE
1.0	0.8924000263214111	job, government, lakh, get, bjp, political, provide, lose, claim, pass	#SadacharYatra The #BJP government had promised employment to 13 lakh in 2007 but only 80,000 got jobs. #SadacharYatra
2.0	0.7767000198364258	https, corruption, co, other, bjp, amp, ever, proof, address, freedom	The Salesman & The Showman. And why BJP must address proof of its own corruption, before accusing others! https://t.co/WjbaK97I92
3.0	0.8062000274658203	power, congress, bjp, work, dmk, divide, anti, fight, donation, rise	เจอกันหาดใหญ่😉 Don Mueang International Airport (DMK) ท่าอากาศยานดอนเมือง http://t.co/HXvAGUM1bC
4.0	0.8446999788284302	seat, bjp, put, trust, share, win, tv, vote, election, pay	These membership drives by BJP shud not be limited 2 just for election purposes. This shud be employed right thru the process. Change system
5.0	0.8942999839782715	co, bjp, election, http, inc, wave, lok_sabha, modi, major, india	Jan 23 : HJS salutes Shiv Sena chief Balasaheb Thackeray on his Birth Anniversary. Read on http://t.co/FklxFvnpq7 http://t.co/C0Xpubdhb7
6.0	0.7050999999046326	time, next, hope, bjp, dalit, mayawati, resignation, together, follower, medium	"***** Hinduism Logic "" GOD BRAHMA MARRIED DAUGHTER SARASWATI & ELEPHANT GOD WAS BORN"" #Atheism #HDL #AAP #BJP http://t.co/MH2uj6g8xk"
7.0	0.9354000091552734	upa, nda, growth, rate, rule, agree, govt, think, woman, economic	#feku thinks #NDA govt was better than UPA. In 9yrs of #UPA economic growth rate avgd 7.9% which was only 6% in #NDA rule. #SadacharYatra
8.0	0.9193000197410583	crore, indian, opinion, bjp, tracker, mean, rajasthan, hold, fall, big	#SadacharYatra The BJP government of Gujarat has managed to give only nine per cent of the total promised jobs. #SadacharYatra
9.0	0.9031000137329102	bjp, pay, show, announce, namoingkp, last, oppose, beat, co, life	#NaMoInGKP - BJP senior leaders will address Vijay Sankalp Rally. LIVE will be available at http://t.co/AWhprQnLB2 #NETAJI
10.0	0.7408999800682068	year, poll, bjp, congress, parliament, try, need, spend, problem, candidate	"Subhash Chandra Bose said ""tum mujhe khoon do main aajadi doonga"". Narendra Modi to Indians ""tum mera saath do main tumhe vikaas doonga"""

Table A.23. (cont'd)

11.0	0.9031000137329102	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	Interesting point Sevanti raises. If the AAP Dharna is anarchy, then what was the demolition of the Babri Masjid by the BJP? Constitutional?
12.0	0.7577999830245972	explain, instead, mp, speech, expose, stop, thing, congress, issue, bjp	AAP confuses majorityism with democracy. BJP presents majorityism as democracy.
13.0	0.8924000263214111	kejriwal, arvind, first, mr, take, co, minister, law, delhi, word	Union Home Minister Sushil Kumar Shinde calls Arvind Kejriwal a mad chief minister. My #cartoon #YEDA http://t.co/I9y2u15MUJ
14.0	0.8062000274658203	bjp, economy, alliance, come, co, cute, break, policy, keep, performance	"MDMK - BJP alliance confirmed. Middle finger for these ""sorinaai""s http://t.co/I9FCqvhhg0"
15.0	0.9031000137329102	india, co, bjp, namo, fund, wrong, listen, avg, today, source	#Modiका विज़न मेरे सपोर्टका रीज़न साबरमतिको साफ़ किया अब गंगा यमुनाकी बारी#NaMoInUP #NaMo4India #BJP #NamoVision http://t.co/Z6Dgb3wVnt
16.0	0.911899983882904	money, want, upa, help, back, refuse, touch, fail, bjp, reason	14) British Govt told Commission that they would not declassify some papers on #Netaji until 2021.UPA did nt help to access these papers
17.0	0.8127999901771545	bjp, talk, become, survey, india, electiontracker, sweep, party, poor, congress	Congress Party as crafted by Mahatma Gandhi was the grandest political formation anywhere in world. Nehru progeny made it private fiefdom.
18.0	0.9031000137329102	amp, bjp, news, gt, party, today, speak, ground, co, expect	11) UPA was hostile towards th Commission.Mukherjee was humiliated for his insistence 2 probe Taiwanese &Russian angles to #Netaji mystery.
19.0	0.8062000274658203	aap, delhi, bjp, co, congress, thank, true, act, non, rti	240 for BJP alone. It has already maximized where it could. I am confident with it, it can make govt. in Delhi.
20.0	0.8616999983787537	modi, co, narendra, congress, deliver, http, face, politic, friend, never	Mani Shankar Aiyyar ko gussa kyun aata hai? Because Narendra Modi chaiwallah is not his cup of tea.
21.0	0.6876999735832214	even, do, call, bjp, supporter, great, many, party, far, dear	Sad that AAP supporters are now the most blind votebank than any other party. BJP is now only party which has supporters who criticise it
22.0	0.8615999817848206	chidambaram, house, pm, leave, shame, history, day, post, co, pc	Ram Jethmalanis letter to P. Chidambaram in NDTV money laundering matter....http://t.co/60COtbob3d

Table A.23. **(cont'd)**

23.0	0.9254999756813049	medium, modi, unite, social, know, narendra, like, co, surge, campaign	50 ppl manage modi social Media: know all- B. G. Mahesh: The techie behind Narendra Modi's campaign - http://t.co/alyo5abKQS #SadacharYatra
24.0	0.8062000274658203	cong, bjp, paidmedia, good, aap, attack, amp, modi, show, stand	2/2 Cong., BJP युवाओं को विदेशी सेक्स और ड्रग्स रैकेट के हवाले करना चाहती हैं, और मीडिया/ पुलिस इस मिलीभगत में शामिल है - देश का दुर्भाग्य
25.0	0.7911999821662903	vote, singh, go, tell, national, rajnath, sonia, choice, pm, president	Reveal truth behind Netajis death, Rajnath Singh tells Centre http://t.co/43DUitVjy4
26.0	0.9354000091552734	give, support, country, upa, lead, bjp, govt, india, right, development	I support BJP led by because Iam worried about stability+development of my country!UPA gave us several scams as Govt of India
27.0	0.8924000263214111	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	Donate Today for Better Tomorrow! Donate online at http://t.co/4SsfQetb5b #Modi4PM http://t.co/y44EoFpnub
28.0	0.9031000137329102	manmohan, singh, bsp, sp, bjp, co, congress, happy, end, massive	RT Massive #SadacharYatra against Guj BJP govt by ends today at Dandi, Bardoli. Completed 4936 kms of tour.
29.0	0.8062000274658203	say, ask, modi, bjp, narendra, seem, chidambaram, co, sc, truth	RT Chidambaram is being economical with truth: BJP http://t.co/8Be0GVoh0L
30.0	0.7386000156402588	people, bjps, leader, see, make, week, bjp, care, eye, top	The point is very simple. The answer to AAPs anarchy is not BJPs anarchy, topped with rabid regressive communalism.

Table A.24. Topic distribution across documents

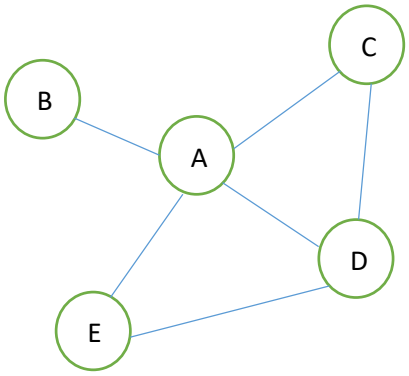
Dominant_Topic	Topic_Keywords	Num_Documents	Perc_Documents
13.0	kejriwal, arvind, first, mr, take, co, minister, law, delhi, word	2312.0	0.0824
14.0	bjp, economy, alliance, come, co, cute, break, policy, keep, performance	389.0	0.0139
20.0	modi, co, narendra, congress, deliver, http, face, politic, friend, never	233.0	0.0083
2.0	https, corruption, co, other, bjp, amp, ever, proof, address, freedom	274.0	0.0098
14.0	bjp, economy, alliance, come, co, cute, break, policy, keep, performance	1666.0	0.0594
11.0	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	1369.0	0.0488
27.0	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	109.0	0.0039
11.0	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	1931.0	0.0688
20.0	modi, co, narendra, congress, deliver, http, face, politic, friend, never	714.0	0.0254
11.0	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	1289.0	0.0459
12.0	explain, instead, mp, speech, expose, stop, thing, congress, issue, bjp	604.0	0.0215
19.0	aap, delhi, bjp, co, congress, thank, true, act, non, rti	520.0	0.0185
27.0	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	179.0	0.0064
27.0	create, co, modi, air, narendra, gorakhpur, bad, set, pm, state	1430.0	0.051
11.0	bjp, dharna, revolution, aap, anarchy, babri_masjid, shri, demolition, constitutional, sevanti_raise	781.0	0.0278

APPENDIX L. Meta-path based Network Embedding

We can set the model as follows, a user *i* belongs to a graph G(V,E), where V is the vertex (node) , E is the edge of this graph and A is the corresponding adjacency matrix. Each user has *i* demographic information, notated as X, called node attributes. The network centrality and community clusters can also be included in X. Each user *i* posts some tweets or retweets on the twitter, which is notated as Y, the label data for each node.

After basic setting, we can transform a graph into a neutral network architectures as follows:

Figure A.14. Adjacency transformation



Vertex set: {A,B,C,D,E}

Adjacency set:

0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	0	1	0

Vertex attributes: X

Label set: {0,1} ep. Tweet, Retweet, etc

The heterogeneous networks is defined as G , where $\mathrm{G} = (\mathrm{V}, \mathrm{E}, \mathrm{T})$,where T is the node type. Two mapping function s, ($\psi: \mathrm{V} \to \mathrm{T_V}, \phi: E \to T_E$). The metapath2vec is a skip-gram style method to represent the network. We want to maximize the probability of context with the given vector v

$$\operatorname{argmax} \sum_{v \in V} \sum_{t \in T_V} \sum_{c_t \in N_t(v)} \log p(c_t | v; \theta)$$

where $N_t(v)$ is the neighbor nodes around given vector v, and type t. The probability is given by a soft-max function:

$$p(c_t|v;\theta) = \frac{e^{Z_{c_t} \cdot Z_v}}{\sum_{u_t \in V_t} e^{Z_{u_t} \cdot Z_v}}$$

Where Z_v means the number v^{th} column of the lower dimension matrix Z

By using negative sampling method, we can reduce the computing burden of this equation and get the following object function

$$\log(Z_{c_t} \cdot Z_v) \sum_{m=1}^{M} E_{u^m} P(u)[\log \sigma(-Z_{u_t} \cdot Z_v)]$$

APPENDIX M. Performance Measure in Recommendation

When we have a classification problem, we want to know how good performance of our model. The basic four measurements are True Positive, True Negative, False Positive, and False Negative. In our research setting, our classes are defined [-1 to -0.5) as class 1, [-0.5 to 0) as class 2, [0 to 0.5) as class 3, and [0.5 to 1] as class 4.

True Positive (TP): The people that truly have the given sentiment scores on the given tweet

True Negative (TN): The people that truly don't have the given sentiment scores on the given tweet

False Positive (FP): The people that truly have the given sentiment scores on the given tweet , but falsely get predicted sentiment scores based on the recommendation

False Negative (FN): The people that truly don't have the given sentiment scores on the given tweet, but falsely get predicted sentiment scores based on the recommendation

Based on these four measurements, we can calculate the Accuracy, Precision, Recall , and F1

Accuracy means how much percentage we truly predict the sentiment scored based on the recommendation algorithm

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Precision means how much percentage we truly predict the sentiment scored given the predicted sentiment scored samples based on the recommendation algorithm

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall means how much percentage we truly predict the sentiment scored given the ground truth sentiment scored samples based on the recommendation algorithm

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1 means the balanced performance between the precision and the recall.

$$\mathrm{F1} = \frac{2 * \mathrm{recall} * \mathrm{precision}}{\mathrm{recall} + \mathrm{precision}}$$

Based on these four measurement, we can continue construct the True Positive Rate, The False Positive Rate, and Area Under the Curve.

True Positive Rate (TPR) means how much percentage we truly predict the sentiment scores given the predicted sentiment scored samples based on the recommendation algorithm. This is as the same as the Recall Ratio

$$\text{True Positive Rate } = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

False Positive Rate (FPR) means how much percentage we don't truly predict the sentiment scores given the false predicted sentiment scored samples based on the recommendation algorithm.

$$\text{True Positive Rate } = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{FN}}$$

Using these two Rates, we can draw a Receiver Operating Characteristic (ROC) curve and the AUC means the Area Under the Receiver Operating Curve. This AUC score can treated as: for each user, how much percentage the truly recommended scores to this focal user compared with the not truly recommended scores to this focal user.

Except the above classification measurement, we also care about the measurement of the recommendation. We choose the Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) in our setting.

Hit Ratio means how much percentage we truly predict the sentiment scores of the focal user based on the recommendation algorithm given the full tweets in the testing sets.

$$\text{HR@K} = \frac{\text{Number of Hits@K Users}}{|\text{Testing Sets of their corresponding Tweets}|}$$

Normalized Discounted Cumulative Gain measures how much percentage the Discounted Cumulative Gains take place of the Ideal Discounted Cumulative Gains.

For example, if we have the list of users n and the list of tweets m , there is a correlation score between each pairs given by the TF-IDF Method in Appendix 11.

The Cumulative Gains at rank position K is calculated as

$$\text{CG}_\text{k} = \sum_{i=1}^{k} correlation_i$$

Where $correlation_i$ means the correlation between each user and tweet pair *i*.

The Discounted Cumulative Gains at rank position K is calculated as

$$\text{DCG}_\text{k} = \sum_{i=1}^{k} \frac{2^{correlation_i} - 1}{\log_2 i + 1}$$

This formula helps to measure the cumulative gains considering their corresponding ranks. The higher of the rank, the more effect of the recommendation.

The Ideal Discounted Cumulative Gains shares the same formula as above but choose the ideal correlation value

$$\text{IDCG}_\text{k} = \sum_{i=1}^{|Ideal\ Correlation|} \frac{2^{correlation_i} - 1}{\log_2 i + 1}$$

Therefore, we can get Normalized Discounted Cumulative Gain (NDCG) as :

$$\text{NDCG}_\text{k} = \frac{DCG_k}{IDCG_k}$$

REFERENCES

REFERENCES

Ahmed, S., & Skoric, M. (2015). Twitter and 2013 Pakistan general election: The case of David 2.0 against goliaths. In Case Studies in e-Government 2.0 (pp. 139-161). Springer, Cham.

Ahmed, S., Jaidka, K., & Cho, J. (2016). The 2014 Indian elections on Twitter: A comparison of campaign strategies of political parties. Telematics and Informatics, 33(4), 1071-1087.

Ahmed, M. A., Lodhi, S. A., & Ahmad, Z. (2017). Political brand equity model: The integration of political brands in voter choice. *Journal of Political Marketing*, *16*(2), 147-179.

Ali, M. Sanni, Rolf H. H. Groenwold, Wiebe R. Pestman, Svetlana V. Belitser, Kit C. B. Roes, Arno W. Hoes, Anthonius de Boer, and Olaf H. Klungel. 2014. "Propensity Score Balance Measures in Pharmacoepidemiology: A Simulation Study." *Pharmacoepidemiology and Drug Safety* 23 (8): 802–11. https://doi.org/10.1002/pds.3574.

Alaa, A. M., Weisz, M., & Van Der Schaar, M. (2017). Deep counterfactual networks with propensity-dropout. *arXiv preprint* arXiv:1706.05966.

Aral, S., Muchnik, L., & Sundararajan, A. (2009). Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proceedings of the National Academy of Sciences*, 106(51), 21544-21549.

Aronow, P. M., & Samii, C. (2017). Estimating average causal effects under general interference, with application to a social network experiment. *The Annals of Applied Statistics*, 11(4), 1912-1947.

Arpino, B., De Benedictis, L., & Mattei, A. (2015). Implementing propensity score matching with network data: The effect of GATT on bilateral trade. Bang H and Robins JM , Doubly robust estimation in missing data and causal inference models, *Biometrics, 61 (2005)*, pp. 962-973.

Austin, Peter C. 2009. "Balance Diagnostics for Comparing the Distribution of Baseline Covariates Between Treatment Groups in Propensity-Score Matched Samples." *Statistics in Medicine* 28 (25): 3083–3107. https://doi.org/10.1002/sim.3697.

Berman, R., Melumad, S., Humphrey, C., & Meyer, R. (2019). A tale of two Twitterspheres: Political microblogging during and after the 2016 primary and presidential debates. *Journal of Marketing Research*, *56*(6), 895-917.

Bermingham, A., & Smeaton, A. (2011). On using Twitter to monitor political sentiment and predict election results. *In Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology*

Bennett, A., & Kallus, N. (2019). Policy evaluation with latent confounders via optimal balance. *arXiv* preprint arXiv:1908.01920.

Bhattacharya, P., Phan, T. Q., Bai, X., & Airoldi, E. M. (2019). A coevolution model of network structure and user behavior: The case of content generation in online social networks. *Information Systems Research*, 30(1), 117-132.

Bobadilla, J., Lara-Cabrera, R., Gonz ́alez-Prieto, Á., & Ortega, F. (2020). Deepfair: deep learning for improving fairness in recommender systems. *arXiv preprint arXiv:2006.05255*.

Bonacich, P. (1987). Power and centrality: A family of measures. *American journal of sociology*, 92(5), 1170-1182.

Borgatti, S. P. (2005). Centrality and network flow. *Social networks*, 27(1), 55-71.

Bose, A., & Hamilton, W. (2019). Compositional fairness constraints for graph embeddings. In *International Conference on Machine Learning* (pp. 715-724). PMLR.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 18–42

Buccoliero, L., Bellio, E., Crestini, G., & Arkoudas, A. (2020). Twitter and politics: Evidence from the US presidential elections 2016. *Journal of Marketing Communications*, 26(1), 88-114.

Bulsara, H. P., & Singh, R. A. (2017) Pre-Election Branding Strategies of Bhartiya Janta Party in Indian General Election 2014: A Content Analysis.

Buyl, M., & Bie, T. D. (2020). Debayes: a bayesian method for debiasing network embeddings. *In International Conference on Machine Learning* (pp. 1220-1229). PMLR.

Buyl, M., & Bie, T. D. (2021). The kl-divergence between a graph model and its fair i-projection as a fairness regularizer. *In Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 351-366). Springer, Cham.

Caliendo, M., & Kopeinig, S. (2008). Some practical guidance for the implementation of propensity score matching. *Journal of economic surveys*, 22(1), 31-72.

Cedric V (2003). *Topics in optimal transportation*. No.58. American Mathematical Soc.

Chang, S., Han, W., Tang, J., Qi, G.-J., Aggarwal, C. C., and Huang, T. S. (2015). Heterogeneous network embedding via deep architectures. *In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM), 119–128

Christakis, N. A., & Fowler, J. H. (2013). Social contagion theory: examining dynamic social networks and human behavior. *Statistics in medicine*, 32(4), 556-577.

Chu, Z., Rathbun, S. L., & Li, S. (2020). Matching in Selective and Balanced Representation Space for Treatment Effects Estimation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (pp. 205-214).

Cortes, C., & Mohri, M. (2014). Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, *519*, 103-126

Current, S., He, Y., Gurukar, S., & Parthasarathy, S. (2022). FairMod: Fair Link Prediction and Recommendation via Graph Modification. *arXiv* preprint arXiv:2201.11596.

Dai, E., & Wang, S. (2021). Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (pp. 680-688).

Dehejia, R. H., & Wahba, S. (2002). Propensity score-matching methods for nonexperimental causal studies. *Review of Economics and statistics*, 84(1), 151-161.

Diwakar, R. (2017). Change and continuity in Indian politics and the Indian party system: Revisiting the results of the 2014 Indian general election. *Asian Journal of Comparative Politics*, 2(4), 327-346.

Dong, Y., Kang, J., Tong, H., & Li, J. (2021). Individual fairness for graph neural networks: A ranking based approach. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (pp. 300-310).

Dong, Y., Chawla, N. V., and Swami, A. (2017). metapath2vec: Scalable representation learning for heterogeneous networks. *In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (ACM)*, 135–144

Dong, Y., Liu, N., Jalaian, B., & Li, J. (2021). EDITS: Modeling and Mitigating Data Bias for Graph Neural Networks. *arXiv* preprint arXiv:2108.05233.

Du, X., Sun, L., Duivesteijn, W., Nikolaev, A., & Pechenizkiy, M. (2019). Adversarial balancing-based representation learning for causal effect inference with observational data. *arXiv preprint* arXiv:1904.13335.

Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference* (pp. 214-226).

Fan, S., Zhu, J., Han, X., Shi, C., Hu, L., Ma, B., & Li, Y. (2019). Metapath-guided heterogeneous graph neural network for intent recommendation. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2478-2486).

Fang, X., Hu, P. J. H., Li, Z., & Tsai, W. (2013). Predicting adoption probabilities in social networks. *Information Systems Research*, 24(1), 128-145.

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., & Venkatasubramanian, S. (2015). Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 259-268).

Forastiere, L., Airoldi, E. M., & Mealli, F. (2020). Identification and estimation of treatment and interference effects in observational studies on networks. *Journal of the American Statistical Association*, 1-18.

Fossen, B. L., Mallapragada, G., & De, A. (2021). Impact of political television advertisements on viewers' response to subsequent advertisements. *Marketing Science*, *40*(2), 305-324.

Fu, T. Y., Lee, W. C., & Lei, Z. (2017). Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. *In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (pp. 1797-1806).

Fu, X., Zhang, J., Meng, Z., & King, I. (2020). Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. *In Proceedings of The Web Conference* 2020 (pp. 2331-2341).

Fu, Z., Xian, Y., Gao, R., Zhao, J., Huang, Q., Ge, Y & De Melo, G. (2020). Fairness-aware explainable recommendation over knowledge graphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 69-78).

Gelper, S., van der Lans, R., & van Bruggen, G. (2021). Competition for attention in online social networks: Implications for seeding strategies. *Management Science*, 67(2), 1026-1047.

Gini, C. (1921). Measurement of inequality of incomes. *The economic journal*, 31(121), 124-126.

Glynn, A. N., & Quinn, K. M. (2010). An introduction to the augmented inverse propensity weighted estimator. *Political analysis*, *18*(1), 36-56.

Gordon, B. R., & Hartmann, W. R. (2013). Advertising effects in presidential elections. *Marketing Science*, *32*(1), 19-35.

Graham, T., Broersma, M., Hazelhoff, K., & Van'T Haar, G. (2013). Between broadcasting political messages and interacting with voters: The use of Twitter during the 2010 UK general election campaign. *Information, communication & society*, 16(5), 692-716.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1), 723-773.

Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864).

Guo, R., Li, J., & Liu, H [a] (2020). Learning individual causal effects from networked observational data. *In Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 232–240.

Guo, R., Li, J., & Liu, H[b] (2020). Counterfactual evaluation of treatment assignment functions with networked observational data. *In Proceedings of the 2020 SIAM International Conference on Data Mining* (pp. 271-279).

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. *In Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 1025-1035).

Hardt, M., Price, E., & Srebro, N. (2016). Equality of opportunity in supervised learning. *Advances in neural information processing systems*, *29*.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020). Lightgcn: Simplifying and powering graph convolution network for recommendation. *In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval* (pp. 639-648).

Heller, R., Rosenbaum, P. R., & Small, D. S. (2010). Using the cross-match test to appraise covariate balance in matched pairs. *The American Statistician*, *64*(4), 299-309.

Hirano, K., & Imbens, G. W. (2004). The propensity score with continuous treatments. *Applied Bayesian modeling and causal inference from incomplete-data perspectives*, 226164, 73-84.

Hill, J. L. (2011). Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1), 217-240.

Ho, Daniel E., Kosuke Imai, Gary King, and Elizabeth A. Stuart. 2007. "Matching as Nonparametric Preprocessing for Reducing Model Dependence in Parametric Causal Inference." *Political Analysis* 15 (3): 199–236. https://doi.org/10.1093/pan/mpl013.

Hong, H., Guo, H., Lin, Y., Yang, X., Li, Z., & Ye, J. (2020). An attention-based graph neural network for heterogeneous structural learning. *In Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 04, pp. 4132-4139).

Hudgens, M. G., & Halloran, M. E. (2008). Toward causal inference with interference. *Journal of the American Statistical Association*, 103(482), 832-842.

Huffman, C., & van Gameren, E. (2018). Covariate balancing inverse probability weights for time-varying continuous interventions. *Journal of Causal Inference*, 6(2).

Imai, K., King, G., & Stuart, E. A. (2014). 8. Misunderstandings Between Experimentalists and Observationalists About Causal Inference. In *Field Experiments and Their Critics* (pp. 196-227). Yale University Press.

Imai, K., & Ratkovic, M. (2014). Covariate balancing propensity score. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, 243-263.

Jackson, M. O., Lin, Z., & Yu, N. N. (2020). Adjusting for peer-influence in propensity scoring when estimating treatment effects. *Available at SSRN* 3522256.

Jaffrelot, C. (2015). The class element in the 2014 Indian election and the BJP's success with special reference to the Hindi belt. *Studies in Indian Politics*, 3(1), 19-38.

Johansson, F., Shalit, U., & Sontag, D. (2016). Learning representations for counterfactual inference. *In International conference on machine learning* (pp. 3020-3029). PMLR.

Johansson, F. D., Shalit, U., Kallus, N., & Sontag, D. (2020). Generalization bounds and representation learning for estimation of potential outcomes and causal effects. *arXiv preprint arXiv:2001.07426*.

Ji, H., Shi, C., & Wang, B. (2018). Attention based meta path fusion for heterogeneous information network embedding. *In Pacific Rim International Conference on Artificial Intelligence* (pp. 348-360). Springer, Cham.

Kallus, N. (2017). A framework for optimal matching for causal inference. In *Artificial Intelligence and Statistics* (pp. 372-381). PMLR.

Kallus N. (2020) DeepMatch: Balancing Deep Covariate Representations for Causal Inference Using Adversarial Training; Proceedings of the 37th *International Conference on Machine Learning*, PMLR 119:5067-5077.

Kang, J., & Tong, H. (2021). Fair Graph Mining. *In Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (pp. 4849-4852).

Kang, K., Park, J., Kim, W., Choe, H., & Choo, J. (2019). Recommender system using sequential and global preference via attention mechanism and topic modeling. In *Proceedings of the 28th ACM international conference on information and knowledge management* (pp. 1543-1552).

Karami, A., Bennett, L. S., & He, X. (2018). Mining public opinion about economic issues: Twitter and the us presidential election. *International Journal of Strategic Decision Sciences*, 9(1), 18-28.

Khare, H. (2015). How Modi Won It: notes from the 2014 election. Hachette India.

Khan, A., Zhang, H., Shang, J., Boudjellal, N., Ahmad, A., Ali, A., & Dai, L. (2020). Predicting Politician's Supporters' Network on Twitter Using Social Network Analysis and Semantic Analysis. *Scientific Programming*, *2020*.

Khan, F. B. (2019). The game of votes: Visual media politics and elections in the digital era. *SAGE Publishing India*.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint* arXiv:1609.02907.

King, G., & Nielsen, R. (2019). Why propensity scores should not be used for matching. *Political Analysis*, *27*(4), 435-454.

Krivitsky, P. N., Handcock, M. S., Raftery, A. E., & Hoff, P. D. (2009). Representing degree distributions, clustering, and homophily in social networks with latent cluster random effects models. *Social networks*, 31(3), 204-213.

Kumar, S., Zhang, X., & Leskovec, J. (2019). Predicting dynamic embedding trajectory in temporal interaction networks. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1269-1278).

Lewis, K., Gonzalez, M., & Kaufman, J. (2012). Social selection and peer influence in an online social network. *Proceedings of the National Academy of Sciences*, 109(1), 68-72.

Li, J., Liu, Y., & Zou, L. (2020). DynGCN: A Dynamic Graph Convolutional Network Based on Spatial-Temporal Modeling. *In International Conference on Web Information Systems Engineering* (pp. 83-95). Springer, Cham.

Li, Y., Tarlow, D., Brockschmidt, M., & Zemel, R. (2017). Gated graph sequence neural networks. *arXiv preprint* arXiv:1511.05493.

Li, S., & Fu, Y. (2017). Matching on balanced nonlinear representations for treatment effects estimation. In *NIPS*.

Lin, J. S., & Himelboim, I. (2019). Political brand communities as social network clusters: winning and trailing candidates in the GOP 2016 primary elections. *Journal of Political Marketing*, *18*(1-2), 119-147.

Liu, L., & Hudgens, M. G. (2014). Large sample randomization inference of causal effects in the presence of interference. *Journal of the american statistical association*, 109(505), 288-301.

Long, M., Wang, J., Ding, G., Sun, J., & Yu, P. S. (2014). Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1410-1417).

Louizos C, Swersky K , Li,YJ, Welling M, and Zemel R. (2016). The variational fair autoencoder. *International Conference on Learning Representations*

Louizos C, Shalit U, Mooij JM, Sontag D, Zemel R, and Welling M, (2017) Causal effect inference with deep latent-variable models, in *Advances in Neural Information Processing Systems*, pp. 6446-6456.

Lu M, Sadiq S, Feaster DJ, Ishwaran H (2018) Estimating individual treatment effect in observational data using random forest methods. *Journal of Computational and Graphical Statistics* 27(1):209–219

Ma L, Krishnan R, Montgomery AL.(2015) Latent Homophily or Social Influence? An Empirical Analysis of Purchase Within a Social Network. *Management Science* 61(2):454-473

Ma, Y., Wang, Y., & Tresp, V. (2020). Causal Inference under Networked Interference. *arXiv preprint* arXiv:2002.08506.

Mallipeddi, R. R., Janakiraman, R., Kumar, S., & Gupta, S. (2021). The Effects of Social Media Content Created by Human Brands on Engagement: Evidence from Indian General Election 2014. *Information Systems Research*.

Manessi, F., Rozza, A., & Manzo, M. (2020). Dynamic graph convolutional networks. *Pattern Recognition*, 97, 107000.

McPherson, M., Smith-Lovin, L., & Cook, J. M. (2001). Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1), 415-444.

Mikolov, T., Chen, K., Corrado, G.S., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *CoRR, abs/1301.3781*.

Mousavi R, Gu B (2019) The Impact of Twitter Adoption on Lawmakers' Voting Orientations. *Information Systems Research* 30(1):133-153

Muja, M., & Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *IEEE transactions on pattern analysis and machine intelligence*, *36*(11), 2227-2240.

Nguyen, G. H., Lee, J. B., Rossi, R. A., Ahmed, N. K., Koh, E., & Kim, S. (2018). Continuous-time dynamic network embeddings. *In Companion Proceedings of the The Web Conference* (pp. 969-976).

Ogburn EL, VanderWeele TJ (2017). Vaccines, contagion, and social networks. *The Annals of Applied Statistics*, 11(2):919–948

Ou, M., Cui, P., Pei, J., Zhang, Z., and Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. *In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (ACM)*, 1105–1114

Ozery-Flato, M., Thodoroff, P., Ninio, M., Rosen-Zvi, M., & El-Hay, T. (2018). Adversarial balancing for causal inference. *arXiv preprint* arXiv:1810.07406.

Panda, A., Kommiya Mothilal, R., Choudhury, M., Bali, K., & Pal, J. (2020). Topical Focus of Political Campaigns and its Impact: Findings from Politicians' Hashtag Use during the 2019 Indian Elections. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1), 1-14.

Pal, J., Chandra, P., & Vydiswaran, V. V. (2016). Twitter and the rebranding of Narendra Modi. *Economic & Political Weekly*, 51(8), 52-60.

Panaretos, V. M., & Zemel, Y. (2019). Statistical aspects of Wasserstein distances. *Annual review of statistics and its application*, 6, 405-431.

Panda, A., Kommiya Mothilal, R., Choudhury, M., Bali, K., & Pal, J. (2020). Topical Focus of Political Campaigns and its Impact: Findings from Politicians' Hashtag Use during the 2019 Indian Elections. *Proceedings of the ACM on Human-Computer Interaction, 4(CSCW1)*, 1-14.

Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., ... & Leiserson, C. (2020). Evolvegcn: Evolving graph convolutional networks for dynamic graphs. *In Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 04, pp. 5363-5370).

Pătruţ, B., & Pătruţ, M. (Eds.). (2014). *Social media in politics: case studies on the political power of social media* (Vol. 13). Springer.

Pearl, J. (2009). *Causality*. Cambridge university press.

Pennacchiotti, M., & Popescu, A. M. (2011). Democrats, republicans and starbucks afficionados: user classification in twitter. *In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 430-438).

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. *In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

Petrova, M., Sen, A., & Yildirim, P. (2021). Social media and political contributions: the impact of new technology on political competition. *Management Science*, *67*(5), 2997-3021.

Pich, C., & Armannsdottir, G. (2018). Political brand image: an investigation into the operationalisation of the external orientation of David Cameron's Conservative brand. *Journal of Marketing Communications*, 24(1), 35-52.

Priya Chacko & Peter Mayer (2014) The Modi lahar (wave) in the 2014 Indian national election: A critical realignment?, *Australian Journal of Political Science*, 49:3, 518-528

Rahman, T., Surma, B., Backes, M., & Zhang, Y. (2019). Fairwalk: Towards fair graph embedding. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. *International Joint Conferences on Artificial Intelligence Organization*, 3289–3295

Rishika R, Ramaprasad J (2019) The Effects of Asymmetric Social Ties, Structural Embeddedness, and Tie Strength on Online Content Contribution Behavior. *Management Science* 65(7):3398-3422.

Rubin, D. B. (2005). Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association* 100(469):322–331

Rutter, R. N., Hanretty, C., & Lettice, F. (2018). Political brands: can parties be distinguished by their online brand personality? *Journal of Political Marketing*, 17(3), 193-212.

Sang, E. T. K., & Bos, J. (2012). Predicting the 2011 dutch senate election results with twitter. *In Proceedings of the workshop on semantic analysis in social media* (pp. 53-60).

Sankar, A., Wu, Y., Gou, L., Zhang, W., & Yang, H. (2020). Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. *In Proceedings of the 13th International Conference on Web Search and Data Mining* (pp. 519-527).

Sardesai, R. (2015). *2014: The election that changed India*. Penguin UK.

Sant'Anna P H.C., Song XJ, Xu X, (2020). "Covariate Distribution Balance via Propensity Scores," *arXiv preprint*, arXiv: 1810.01370

Schneiker, A. (2019). Telling the story of the superhero and the anti-politician as president: Donald Trump's branding on Twitter. *Political studies review*, 17(3), 210-223.

Schwab, P., Linhardt, L., & Karlen, W. (2019). Perfect match: A simple method for learning representations for counterfactual inference with neural networks. *arXiv preprint* arXiv:1810.00656.

Seo, Y., Defferrard, M., Vandergheynst, P., & Bresson, X. (2018). Structured sequence modeling with graph convolutional recurrent networks. *In International Conference on Neural Information Processing* (pp. 362-373). Springer, Cham.

Shalit, U., Johansson, F. D., & Sontag, D. (2017). Estimating individual treatment effect: generalization bounds and algorithms. *In International Conference on Machine Learning* (pp. 3076-3085). PMLR.

Shalizi, C. R., & Thomas, A. C. (2011). Homophily and Contagion Are Generically Confounded in Observational Social Network Studies. *Sociological Methods & Research*.;40(2):211-239

Sharma, A., Gupta, G., Prasad, R., Chatterjee, A., Vig, L., & Shroff, G. (2020). MultiMBNN: Matched and Balanced Causal Inference with Neural Networks. *arXiv preprint* arXiv:2004.13446.

Skoric, M., Poor, N., Achananuparp, P., Lim, E. P., & Jiang, J. (2012). Tweets and votes: A study of the 2011 singapore general election. *In 2012 45th hawaii international conference on system sciences* (pp. 2583-2591). IEEE

Shi, C., Blei, D. M., & Veitch, V. (2019). Adapting neural networks for the estimation of treatment effects. *arXiv preprint* arXiv:1906.02120.

Shi, C., Hu, B., Zhao, W. X., & Philip, S. Y. (2018). Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 31(2), 357-370.

Song, L., Langfelder, P., & Horvath, S. (2012). Comparison of co-expression measures: mutual information, correlation, and model based indices. *BMC bioinformatics*, 13(1), 1-21.

Song, T., Tang, Q., & Huang, J. (2019). Triadic Closure, Homophily, and Reciprocation: An Empirical Investigation of Social Ties Between Content Providers. *Information Systems Research* 30(3):912-926

Spinelli, I., Scardapane, S., Hussain, A., & Uncini, A. (2021). FairDrop: Biased Edge Dropout for Enhancing Fairness in Graph Representation Learning. *IEEE Transactions on Artificial Intelligence*.

Susarla, A., Oh, J. H., & Tan, Y. (2016). Influentials, imitables, or susceptibles? Virality and word-of-mouth conversations in online social networks. *Journal of Management Information Systems*, 33(1), 139-170.

Sweet, T., Adhikari, S. (2020) A Latent Space Network Model for Social Influence. *Psychometrika* 85, 251–274

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). Line: Large-scale information network embedding. *In International Conference on World Wide Web*. 1067–1077

Tchetgen, E. J. T., & VanderWeele, T. J. (2012). On causal inference in the presence of interference. *Statistical methods in medical research*, 21(1), 55-75.

Velickovic, P., Cucurull, G., Casanov a, A., Romero, A., Lio, P., and Bengio, Y. (2018). Graph attention networks *International Conference on Learning Representations (ICLR)*

Veitch, V., Wang, Y., & Blei, D. M. (2019). Using embeddings to correct for unobserved confounding in networks. *arXiv preprint arXiv:1902.04114*.

Wang, Y., Lewis, M., & Schweidel, D. A. (2018). A border strategy analysis of ad source and message tone in senatorial campaigns. *Marketing Science*, *37*(3), 333-355.

Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., & Yu, P. S. (2019). Heterogeneous graph attention network. *In The World Wide Web Conference (pp. 2022-2032).*

Wang, N., Lin, L., Li, J., & Wang, H. (2021). Unbiased Graph Embedding with Biased Graph Observations. *arXiv* preprint arXiv:2110.13957.

Wang, C., Song, Y., Li, H., Zhang, M., & Han, J. (2018). Unsupervised meta-path selection for text similarity measure based on heterogeneous information networks. Data Mining and Knowledge Discovery, 32(6), 1735-1767.

Wei, X., Liu, Z., Sun, L., & Yu, P. S. (2018). Unsupervised meta-path reduction on heterogeneous information networks. arXiv preprint arXiv:1810.12503.

Wu, L., Sun, P., Hong, R., Fu, Y., Wang, X., & Wang, M. (2018). Socialgcn: An efficient graph convolutional network based model for social recommendation. *arXiv preprint* arXiv:1811.02815.

Wu, Q., Zhang, H., Gao, X., He, P., Weng, P., Gao, H., & Chen, G. (2019). Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. *In The World Wide Web Conference* (pp. 2091-2102).

Wu, L., Chen, L., Shao, P., Hong, R., Wang, X., & Wang, M. (2021). Learning fair representations for recommendation: A graph-based perspective. In *Proceedings of the Web Conference 2021* (pp. 2198-2208).

Xie, R., Liu, Z., Jia, J., Luan, H., & Sun, M. (2016). Representation learning of knowledge graphs with entity descriptions. *In Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 30, No. 1).

Xue, H., Yang, L., Jiang, W., Wei, Y., Hu, Y., & Lin, Y. (2020). Modeling dynamic heterogeneous network for link prediction using hierarchical attention with temporal rnn. *arXiv preprint* arXiv:2004.01024.

Yao, L., Li, S., Li, Y., Huai, M., Gao, J., & Zhang, A. (2018). Representation learning for treatment effect estimation from observational data. In: *Advances in Neural Information Processing Systems*, pp 2634–2644

Yao, L., Li, S., Li, Y., Xue, H., Gao, J., & Zhang, A. (2019, August). On the estimation of treatment effect with text covariates. *In International Joint Conference on Artificial Intelligence.*

Yun, S., Jeong, M., Kim, R., Kang, J., & Kim, H. J. (2019). Graph transformer networks. *Advances in Neural Information Processing Systems*, 32, 11983-11993.

Zeng, Z., Islam, R., Keya, K. N., Foulds, J., Song, Y., & Pan, S. (2021). Fair representation learning for heterogeneous information networks. *arXiv preprint arXiv:2104.08769*.

Zhang, B., Pavlou, P. A., & Krishnan, R. (2018). On Direct vs. Indirect Peer Influence in Large Social Networks. *Information Systems Research* 29(2):292-314

Zhang, L., Fu, J., Wang, S., Zhang, D., Dong, Z., & Chen, C. P. (2019). Guide subspace learning for unsupervised domain adaptation. *IEEE transactions on neural networks and learning systems*, *31*(9), 3374-3388.

Zhang, C., Song, D., Huang, C., Swami, A., & Chawla, N. V. (2019). Heterogeneous graph neural network. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 793-803).

Zhang, L., & Chung, D. J. (2020). The Air War vs. the Ground Game: An Analysis of Multichannel Marketing in US Presidential Elections. *Marketing Science*, *39*(5), 872-892.

Zheng, J., Qi, Z., Dou, Y., & Tan, Y. (2019). How Mega Is the Mega? Exploring the Spillover Effects of WeChat Using Graphical Model. *Information Systems Research* 30(4):1343-1362.